Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Luka Kacil

Sistem za analizo sentimenta v komentarjih o mobilnih aplikacijah

diplomsko delo

univerzitetni študijski program računalništvo in informatika

Mentor: izr. prof. dr. Zoran Bosnić

Ljubljana, 2016

Besedilo je oblikovano z urejevalnikom besedil LaTeX.

Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Tematika naloge:

V diplomskem delu naj kandidat predstavi področje analize sentimenta komentarjev na spletu in naj opravi kratek pregled obstoječih metod za to nalogo. V nadaljevanju naj predlaga svoj sistem za analizo spletnih komentarjev o aplikacijah iz spletne trgovine Google Play. Po predstavitvi in preliminarni analizi problemske domene naj predlaga atribute in kombinacijo pristopov strojnega učenja, ki omogočajo uvrščanje komentarjev v različne razrede. Svoj pristop naj eksperimentalno ovrednoti in ga primerja z drugimi obstoječimi pristopi.

Zahvaljujem se izr. prof. dr. Zoranu Bosniću za nenehno vzpodbudo, nasvete in pomoč pri izdelavi diplomske naloge.
Zahvaljujem se tudi svoji družini za podporo in razumevanje tekom študija.

Kazalo

Povzetek

Abstract

1 Uvod **1**

2 Analiza sentimenta **5**

2.1 Definicija problema 6

2.2 Cilji in naloge 8

2.3 Označevanje na nivoju dokumenta 9

2.4 Nivo povedi 13

2.5 Nivo entitete in aspekta 14

2.6 Leksikoni mnenjskih besed 15

2.7 Orodja za analizo sentimenta 17

3 Analiza vhodnih podatkov **21**

3.1 Dolžina komentarjev 21

3.2 Sestava besed v komentarjih 22

3.3 Pravilnost črkovanja in nebesedni znaki 23

3.4 Porazdelitev komentarjev glede na oceno 23

3.5 Izbor komentarjev 26

4 Analiza sentimenta s pomočjo nadzorovanega učenja **27**

4.1 Razčlenjevanje 28

4.2 Normalizacija in negacija 29

4.3 N-grami 31
4.4 Frekvenca pojavitev žetona 32
4.5 Korenjenje 34
4.6 Popravljanje črkovanja 35
4.7 Oblikoslovno označevanje besedila 37
4.8 Klasifikacija z nadzorovanim učenjem 38
4.9 Vrednotenje klasifikatorjev 47
4.10 Predstavitev in izbor atributov 51
4.11 Rezultati 56

5 Primerjava z drugimi sistemi **63**
5.1 VADER 63
5.2 Indico 64
5.3 Ostali sistemi 66
5.4 Rezultati 66

6 Zaključek **69**

Seznam slik **72**

Seznam tabel **74**

Literatura **77**

Seznam uporabljenih kratic

kratica	**angleško**	**slovensko**
CA	classification accuracy	klasifikacijska točnost
NB	naive Bayes	naivni Bayes
SVM	support vector machine	metoda podpornih vektorjev
LR	logistic regression	logistična regresija
PMI	pointwise mutual information	točkovna medsebojna informacija
JSON	JavaScript object notation	objektna notacija JavaScript
TF-IDF	term frequency–inverse	obratna frekvenca pojavljanja
	document frequency	izraza v zbirki besedil
ASCII	American Standard Code	ameriški standardni nabor
	for Information Interchange	za izmenjavo informacij
χ^2	chi-squared	hi kvadrat

Povzetek

Naslov: Sistem za analizo sentimenta v komentarjih o mobilnih aplikacijah

Cilj diplomske naloge je bil implementirati sistem za označevanje komentarjev, ki izražajo navdušenje v spletni trgovini Google Play. Pri tem smo najprej opravili pregled področja analize sentimenta, nato pa analizirali komentarje in se bolje spoznali s problemsko domeno. Opisali smo teoretično podlago vseh metod, ki smo jih nato uporabili pri gradnji sistema. Najprej smo vhodne komentarje pretvorili v žetone besed in jih normalizirali, negirali in iz njih ustvarili n-grame. Nato smo uporabili korenjenje, popravljanje črkovanja, dodajanje oblikoslovnih oznak in dodatnih zunanjih atributov in ustvarili osem različnih naborov atributov. Iz vsakega nabora smo izbrali najboljše atribute s pomočjo metode χ^2. Za klasifikacijo smo nato uporabili modele, kot so naivni Bayes, logistična regresija in metoda podpornih vektorjev. Sledilo je ovrednotenje klasifikatorjev s pomočjo notranjega prečnega preverjanja, klasifikacijske točnosti, priklica, preciznosti, mere F1 in statističnih testov. Na koncu smo označevanje komentarjev iz naše problemske domene testirali na obstoječih rešitvah za analizo sentimenta in primerjali rezultate. Ugotovili smo, da obstajajo statistično pomembne razlike med rezultati klasifikatorjev. Prav tako so obstajale statistično pomembne razlike med rezultati nekaterih naborov atributov. Ugotovili smo tudi, da obstajajo statistične pomembne razlike med rezultati obstoječih rešitev in nekaterih naših modelov.

Ključne besede: analiza sentimenta, nadzorovano strojno učenje, metoda podpornih vektorjev, naivni Bayes, logistična regresija.

Abstract

Title: System for sentiment analysis of comments about mobile applications

The goal of this thesis was to build a sentiment analysis system, which can tag exuberant reviews in the Google Play store. First we gave an overview of the sentiment analysis field and analysis of input comments to better understand our problem domain. We described theoretical foundations of every method used to build our system. We started by transforming input reviews into tokens which were then normalized, negated and transformed in n-grams. After that we used stemming, spell correction, part of speech tagging and adding other attributes to generate eight different collections of features. We selected best features from every collection with χ^2 method. For classification we used naive Bayes, logistic regression and support vector machine to classify reviews. After that we evaluated classifiers by using internal cross-validation and computing classification accuracy, recall, precision, F1 score and statistical tests. In the end we tested tagging reviews from our problem domain with existing solutions for sentiment analysis and compared the results. Results revealed that there were statistically significant differences between classifiers. There were also statistically significant differences between some feature collections. Results also revealed that there were statistically significant differences between existing solutions and some of our models.

Keywords: sentiment analysis, supervised machine learning, support vector machine, naive Bayes, logistic regression.

Poglavje 1

Uvod

Količina uporabniško zgeneriranih vsebin na spletu v zadnjem desetletju raste z nepredstavljivo hitrostjo in je kot taka na voljo vsem. Ljudje uporabljajo forume, bloge, socialna omrežja in ostale komunikacijske kanale, da delijo svoja mnenja z ostalimi. Dandanes ljudje pred nakupom novega produkta najprej preverijo, kaj drugi menijo o njem. To na koncu v veliki meri vpliva na končno odločitev o nakupu. Pred eksplozijo vseh teh prosto dostopnih vsebin smo bili primorani iskati mnenja na bolj konvencionalen način. Za mnenje smo povprašali prijatelje, sorodnike in ostale ljudi, s katerimi interaktiramo na dnevni ravni. Danes obstajajo strani, kot so Amazon.com, ki ponujajo različne produkte, ob tem pa agregirajo in prikazujejo povratne informacije od ljudi, ki so artikel že kupili. S tem je bodočemu kupcu omogočen preprost pregled nad tisočimi mnenji, ki so lahko razdeljena v nekaj zelo preprostega, kot so pozitivna in negativna mnenja ali pa bolj kompleksne strukture, kjer so uporabniki posebej ocenili različne lastnosti nekega produkta.

Na drugi strani podjetja zanima, kaj o njihovih produktih menijo uporabniki. Z zajemom ogromnih količin mnenj lahko analizirajo priljubljenost znamke in povzetke uporabijo pri modeliranju tržnih strategij in nadaljnjem razvoju prihodnjih produktov. Interes za analizo teh vsebin seveda ni omejen samo na prej opisane primere in ga najdemo tudi na drugih področjih. Zbiranje in analiza sentimenta se tako uporablja pri razumevanju svetovnih

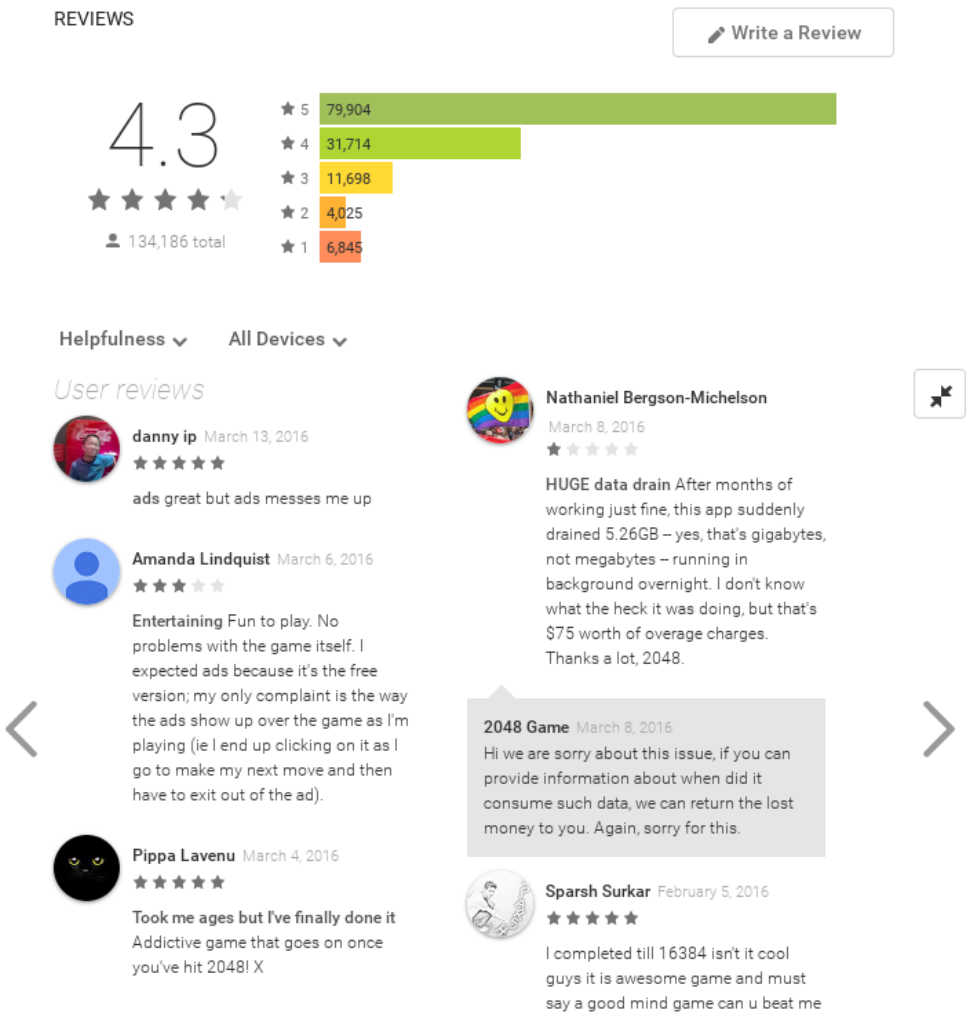
dogodkov kot tudi za razumevanje javnega mnenja v različnih političnih temah. Analiziranje mnenj uporabnikov se uporablja pri pozicioniranju oglasov na spletu in za detektiranje in odstranjevanje sovražnega govora na socialnih omrežjih.

Zato ni nepričakovano, da je nastal velik interes za analizo sentimenta iz nestrukturiranih virov. Analiza sentimenta spada na raziskovalno področje analize besedil v naravnem jeziku, kjer se osredotoča na to, kako so mnenja, odnosi, čustva in posameznikova perspektiva izraženi skozi jezik. Pri analizi sentimenta imamo pester nabor orodij in tehnik, ki nam omogočajo izluščiti in ovrednotiti subjektivne informacije v ogromnih podatkovnih bazah, ki so danes na voljo.

V okviru diplomske naloge smo pogledali, kako se področje analize sentimenta trenutno razvija in kateri pristopi se uporabljajo. V ta namen smo izbrali nekaj prosto dostopnih in komercialnih orodij. Nato smo določili domeno problema, ki obsega angleške komentarje mobilnih aplikacij v spletni trgovini Google Play Store, kot je to prikazano na sliki 1.1. Zanimalo nas je, kakšno mnenje imajo uporabniki o aplikaciji, in poskušali smo identificirati komentarje, ki izražajo navdušenje. Ti izredno pozitivni komentarji nas zanimajo zato, ker so vhodna točka za posredno ocenjevanje širjenja aplikacije med uporabniki. To preprosto pomeni, da čim bolj kot je uporabnik navdušen nad aplikacijo, tem bolj verjetno je, da jo bo priporočil svojim prijateljem. Razvili smo sistem, ki temelji na nadzorovanem učenju, in ga na koncu primerjali z že razvitimi sistemi na naši problemski domeni.

Diplomsko nalogo smo razdelili na 4 poglavja. V prvem poglavju naredimo pregled nad področjem analize sentimenta in znotraj njega izberemo že razvita prosto dostopna oziroma komercialna orodja, ki jih bomo uporabili za primerjavo z našim sistemom. V drugem poglavju opišemo problemsko domeno in naredimo analizo komentarjev za boljše razumevanje vhodnih podatkov. V tretjem poglavju se osredotočimo na gradnjo našega sistema. Poglavje je razdeljeno na več podpoglavij. Najprej pogledamo problem razčlenjevanja besedilnih dokumentov, negacije, in generiranja n-gramov. Nato

opišemo bolj napredne metode za obdelavo besedil, kot so korenjenje, popravljanja črkovanja in oblikoslovno označevanje besedila. V nadaljevanju opišemo izbrane klasifikatorje in njihovo vrednotenje uspešnosti. Nato določimo različne modele atributov in metodo za izbor najboljših. Na koncu primerjamo izbrane klasifikatorje in modele atributov. V zadnjem poglavju primerjamo naš sistem z ostalimi na naši problemski domeni.



Slika 1.1: Prikaz komentarjev v trgovini Google Play pod eno izmed aplikacij.

Poglavje 2

Analiza sentimenta

Analiza sentimenta spada na področje procesiranja naravnega jezika in je postala zelo aktualna šele v zadnjem desetletju. Pred tem je bil obseg digitalnih vsebin zelo omejen in šele z vzponom Web 2.0 se je pokazal interes za njegovo poglobljeno raziskovanje. Področje se ukvarja z identifikacijo in ekstrakcijo subjektivnih informacij iz nestrukturiranega teksta.

Najbolj osnoven pristop je označevanje besedila kot pozitivnega ali negativnega. Na primer: *"I really enjoyed the movie!"* je primer besedila, ki bi ga označili kot pozitivnega. Vendar pa postane avtomatsko in natančno označevanje težek problem, ker se niti ljudje popolnoma ne strinjamo glede mnenj, ki so izražena v besedilih. Podjetje Biz360 je naredilo analizo sentimenta ročno označenih dokumentov in je ugotovilo, da so se njihovi ocenjevalci strinjali le v 79% primerih[1]. Poleg tega, besedila zapisana v naravnem jeziku, skrivajo veliko kompleksnih lastnosti, kot so sarkazem in idiomi. Prav tako je poznavanje domene oziroma ciljne entitete, o kateri je bilo izraženo mnenje, zelo pomembno. Kot lahko vidimo v primeru:

"I suggest that you read the book.",

bi bil ta komentar smatran za pozitivnega, če ga beremo na strani, kjer ljudje ocenjujejo knjige. Če pa ga najdemo na strani, kjer se ocenjuje filme, bi ta komentar najverjetneje spadal med negativne.

[1] http://goo.gl/vRGcuA

Pregled nad akademskim razvojem področja je bil že obsežno opisan v literaturi [13, 17, 19, 21]. Skozi ta dela je bil problem analize sentimenta tudi bolj teoretično definiran in te definicije so predstavljene v naslednjih razdelkih.

2.1 Definicija problema

Pri analizi sentimenta nas poleg izraženega mnenja zanima, kdo to mnenje izraža, kdaj ga izraža in o čem je bilo mnenje izraženo.

> *"Me and Alice bought smartphones in BestBuy yesterday. She got an iPhone and I bought a Nexus. I am very impressed by the speed but battery life isn't too good. Alice is very satisfied with the photo quality. When we compared cameras it was pretty clear her device makes pictures with better contrast."* (A)

V zgornjem primeru (A) vidimo, da gre pri komentarju za izraz več mnenj o različnih lastnostih dveh objektov. Poleg tega so izražena mnenja dveh oseb: osebe, ki je komentar napisala, in osebe Alice.

Pri tem definiramo naslednje lastnosti problemske domene, ki jih to področje obravnava:

Izražalec mnenja: označimo ga z oznako h in predstavlja osebo, ki podaja svoje mnenje o neki entiteti.

Časovna komponenta: označuje, kdaj je bilo mnenje podano. Velikokrat nas zanima, kako se mnenje o nekem produktu ali storitvi spreminja skozi čas. Označili jo bomo s t.

Objekt in njegovi atributi: objekt je ciljna entiteta, o kateri podajamo mnenje. Objekt lahko ima več atributov, na katere se mnenje nanaša.

V zgornjem primeru (A) imamo dva objekta: Nexus in iPhone. Ocenjeni atributi Nexusa so: hitrost, življenjska doba baterije in kvaliteta slike. Ocenjeni atribut iPhone-a pa je kvaliteta slike. Objekte bomo označili z oznako o, njihove atribute pa z oznako f. Atributi se lahko nadalje delijo na eksplicitne in implicitne. Eksplicitni so tisti, ki so v dokumentu direktno navedeni, implicitni pa tisti, kjer atribut ni prisoten, vendar skozi kontekst besedila razumemo, da se nanaša nanj. *"This camera is too heavy",* je primer, kjer atribut teža ni naveden, vendar skozi kontekst razumemo, da se komentar nanaša nanj.

Mnenje in orientacija: mnenje o nekem objektu ali njegovih atributih je izraženo skozi orientacijo, ki je lahko pozitivna ali negativna. V primeru (A) je v drugi, tretji in zadnji povedi izraženih več mnenj o kupljenih napravah. Mnenja nadalje delimo na:

Neposredna: neposredna mnenja se nanašajo direktno na objekt oziroma njegove atribute. V primeru (A) so neposredna mnenja izražena v tretji in četrti povedi.

Primerjalna: primerjalna mnenja vsebujejo dva ali več objektov, ki jih med seboj primerjamo po različnih atributih. V zgornjem primeru (A) imamo v zadnjem stavku izraženo primerjalno mnenje.

Model objekta: objekt o je predstavljen s končno množico atributov $F = \{f_1, f_2, f_3, ..., f_n\}$ in ta množica ponavadi vsebuje tudi objekt o kot poseben atribut. Vsak atribut $f_i \in F$ je nato izražen s končno množico besed ali fraz $W_i = \{w_1, w_2, w_3, ..., w_n\}$, ki so sinonimi f_i. Tako je na primer atribut f_i, ki opisuje zaslon, izražen s sinonimi kot so: "screen", "display" in "monitor". Ti sinonimi so elementi množice W_i.

Model mnenjskega dokumenta: mnenjski dokument d vsebuje mnenja izražalcev mnenj $h_1, h_2, ..., h_n$ o objektih $o_1, o_2, o_3, ..., o_n$. Mnenja o objektih so izražena na podmnožici atributov teh objektov.

2.2 Cilji in naloge

Cilj analize sentimenta pri podanem mnenjskem dokumentu d je najti vse petorke $(o_i, f_{ij}, s_{ijkl}, h_k, t_l)$ v d, kjer o_i predstavlja enega izmed objektov, f_{ij} enega izmed atributov tega objekta, h_k predstavlja osebo, ki izraža mnenje v času t_l in s_{ijkl} predstavlja zaznani sentiment.

Treba je poudariti, da vedno ne določamo vseh elementov petork, ker včasih vnaprej poznamo izražalca mnenja, časovno komponento in tudi objekt. Na primer, objave na forumih imajo ponavadi vnaprej določenega avtorja in tudi časovna komponenta je znana. Poleg tega je pomembno poudariti tudi, da izraženost mnenja ni vedno binarni atribut. Tako so nekatera mnenja izražena močneje kot druga. *"This headphones suck!"* izraža izredno negativno mnenje, medtem ko je *"I don't particularly like the color of these headphones."* sicer še zmeraj negativno, vendar je intenziteta veliko manjša. Zgornja definicija petorke služi kot shema, ki nam omogoča spremeniti nestrukturiran dokument v strukturirano mnenjsko podatkovno obliko.

Z določenim ciljem in definiranimi lastnostmi iz razdelka 2.1, razdelimo naloge analize sentimenta v 6 sklopov, kot sledi v nadaljevanju.

Ekstrakcija objekta in razvrščanje: v tem koraku iz mnenjskega dokumenta d izluščimo vse objekte o in njihove sinonime ter jih razvrščamo v gruče.

Ekstrakcija atributov in razvrščanje: v mnenjskem dokumentu d poiščemo atribute predhodno določenih objektov. Nato za atribute F_i posameznega objekta o_i poiščemo sinonime in jih razvrščamo.

Določanje izražalca mnenja: iz d izluščimo osebe, ki podajajo mnenja. Korak je enak prejšnjima in se konča s razvrščanjem oseb.

Določanje časovne komponente: v tem koraku izluščimo informacije o času, ki nam povedo, kdaj so bila mnenja izražena.

Označevanje sentimenta: določimo, ali je mnenje o nekem atributu fij pozitivno, negativno ali nevtralno.

Generacija mnenjske petorke: na koncu generiramo vse petorke $(o_i, f_{ij}, s_{ijkl}, h_k, t_l)$.

Analiza sentimenta, ki v celoti sledi tem korakom, je navadno označena za analizo sentimenta na nivoju aspekta (*aspect-based sentiment analysis*).

Označevanje sentimenta v razdelimo na tri različne nivoje: nivo dokumenta, nivo povedi in nivo entitete in aspekta. Eden najbolj raziskanih pristopov je klasifikacija na nivoju celotnega dokumenta [22,25]. Na tem nivoju obravnavamo celoten mnenjski dokument kot osnovno enoto informacije in ga označimo kot negativnega ali pozitivnega. Na nivoju povedi je osnovna enota informacije poved. Metoda vsebuje dodatni korak, ki loči mnenjske povedi od povedi, ki mnenj ne vsebujejo. Zadnji nivo, ki je trenutno tudi najbolj aktualen, je nivo entitete in aspekta.

2.3 Označevanje na nivoju dokumenta

Označevanje na nivoju dokumenta razdelimo na dva različna pristopa. V prvem jo rešimo z nadzorovanim učenjem, v drugem pa s statistično analizo zaporedja besed. Akademsko raziskovanje obeh pristopov je bilo že zelo pokrito, kar ju uvršča med najbolj opisane in raziskane [24].

Definicija problema in predpostavke: pri danem mnenjskem dokumentu d in objektu o, želimo s strani izražalca mnenja h določiti sentiment s v nekem času t. Pri tem nam je vnaprej poznan izražalec mnenja, objekt, o katerem se mnenje izraža, in kdaj je bilo to mnenje izraženo. Prav tako predpostavljamo, da je mnenje v dokumentu izraženo s strani ene osebe, in se nanaša na objekt kot celoto. Zaradi tega lahko petorko poenostavimo v $(-, SPLOSNO, s, -, -)$, kjer vpeljemo oznako $SPLOSNO$, ki zajema vse atribute objekta. Ker so objekt, izražalec mnenja in čas vnaprej znani in fiksni, niso več del našega problema in jih v petorki označimo z $-$. Tako postane problem gradnje petorke določanje sentimenta s.

Če označimo mnenje kot negativno ali pozitivno, potem gre pri tem pristopu za binarno označevanje, če pa želimo določiti s na podlagi numeričnih vrednosti, je problem regresijski. Problem označevanja na nivoju dokumenta postane, če naše predpostavke ne držijo, in je v mnenjskem dokumentu izraženih več mnenj o različnih atributih objekta, ali pa so v dokumentu podana mnenja več kot ene osebe. Zato se ta pristop ponavadi uporablja pri analizi sentimenta krajših mnenjskih dokumentov, kot so komentarji o različnih produktih in storitvah, ki so napisani s strani ene osebe. Pri mnenjskih dokumentih, kot so blogi, pa ta pristop ni več tako praktičen, ker so besedila daljša in lahko izražajo mnenja s strani različnih oseb o različnih objektih.

V naslednjih razdelkih pogledamo označevanje komentarjev s pomočjo nadzorovanega učenja in statistične analize zaporedja besed.

2.3.1 Označevanje s pomočjo nadzorovanega učenja

Najbolj popularen pristop označevanja na nivoju celotnega mnenjskega dokumenta je nadzorovano učenje. Pri tem mnenjski dokument uvrstimo v pozitiven ali negativen razred. Uporabimo lahko tudi tretji razred, ki je nevtralen in v katerega uvrstimo besedila brez zaznanega sentimenta. Za grajenje učne in testne množice si lahko pomagamo s spremljajočo oceno, ki je ponavadi prisotna pri komentarjih o različnih produktih ali storitvah. Lahko pa komentarje za učenje označimo tudi ročno. Tako postane označevanje mnenjskega dokumenta problem klasifikacije tekstovnega dokumenta. Tradicionalni pristopi pri označevanju teksta se osredotočajo na klasifikacijo v razrede glede na temo dokumenta, kot so šport, politika, kultura, nezaželena sporočila in podobno. Vendar je Christopher je v [4] prikazal, da je klasifikacija sentimenta precej težji problem. Kjer je klasifikacija nezaželene pošte in določanje teme besedila že dobro rešen problem, klasifikacija sentimenta na realnih problemih ponavadi ne presega 90% točnosti[2]. To pa predvsem zato, ker je sentiment v besedilih izražen skozi kompleksne lingvistične strukture.

Pristop h klasifikaciji s pomočjo nadzorovanega učenja razdelimo na več

[2]http://sentiment.christopherpotts.net/lingcog.html

segmentov, cilj vsakega ali skupine uporabljenih segmentov pa je izbrati najbolj primeren vektor atributov, ki ga potem prejme klasifikator.

Izrazi in njihova frekvenca (angl. *terms and frequency*): izrazi so lahko posamezne besede mnenjskega besedila ali pa njihovi n-grami. Pri grajenju vektorja atributov lahko uporabimo različne vrednosti za posamezen izraz. Te segajo od binarne vrednosti, ki predstavlja prisotnost ali neprisotnost izraza, frekvence pojavljanja izraza v besedilu do inverzne frekvence pojavljanja izraza v zbirki besedil (TF-IDF).

Oblikoslovno označevanje besedila (angl. *POS tagging*): pri tem besede označimo s skladenjskimi oznakami, ki nam omogočajo, da so nekatere besede ali fraze obravnavane drugače (na primer pridevniki).

Mnenjske besede in fraze: mnenjske besede, ki izražajo sentiment, so večinoma pridevniki in prislovi (*excellent, poor, nice*) in tudi samostalniki. Med mnenjske fraze spadajo tudi idiomi (*"cut corners."*).

Pravila mnenj: poleg mnenjskih besed in fraz obstajajo tudi drugi besedilni konstrukti, ki izražajo sentiment ali nakazujejo mnenje. Na primer v povedi *"After drinking that energy drink, my heart rate went up to 120."*, sprememba srčnega utripa izven normalnega območja nakazuje nezaželen učinek energijske pijače in zaradi tega poved izraža negativno mnenje.

Negacija: sem spadajo vse besede in fraze, ki obrnejo orientacijo mnenja. *"I don't like this coffee"*, je primer, kjer beseda *don't* obrne orientacijo pridevnika *like*.

Sintaktične odvisnosti: v ta razdelek spadajo vse tehnike, ki generirajo vektorje atributov na podlagi odvisnostnih lastnosti besed v dokumentu.

Potem ko zgeneriramo vektor atributov, lahko uporabimo eno od standardnih metod strojnega učenja za klasifikacijo.

2.3.2 Označevanje s pomočjo statistične analize zaporedja besed

V enem izmed prvih pristopov Turney [25] opiše metodo, kako razvrstiti mnenjske dokumente na pozitivne in negativne s pomočjo referenčnih besed "excellent" in "poor", sintaktičnih pravil in analize, ki izkorišča točkovno medsebojno informacijo besed (angl. *pointwise mutual information*).

Korak 1: iz besedila izluščimo vse fraze, ki vsebujejo pridevnike in prislove in sledijo vnaprej podanim oblikoslovnim pravilom. Algoritem izlušči dve zaporedni besedi, kjer mora biti ena izmed besed pridevnik ali prislov, druga beseda pa določa kontekst besedila. Ti besedi potem izberemo, če njuno oblikoslovno oznako najdemo v tabeli 2.1. Na primer, vrstica štiri označuje samostalnik za prvo besedo in pridevnik za drugo. Tretja beseda (ki ni izluščena), pa ne sme biti samostalnik.

Korak 2: Izračunamo mnenjsko orientacijo s posamezne izluščene fraze, glede na njeno razmerje s pozitivno referenčno besedo *"excellent"* in negativno referenčno besedo *"poor"*:

$$s(fraza) = PMI(fraza, "excellent") - PMI(fraza, "poor") \quad (2.1)$$

Razmerje s pozitivno in negativno referenčno besedo izračunamo s pomočjo mere PMI (angl. *pointwise mutual information, točkovna medsebojna informacija*):

$$PMI(fraza, referenca) = log_2 \left(\frac{P(fraza \cap referenca)}{P(fraza)P(referenca)} \right) \quad (2.2)$$

$P(fraza \cap referenca)$ predstavlja verjetnost, da se $fraza$ in $referenca$ pojavita skupaj, in $P(fraza)P(referenca)$ verjetnost, da se ta dva izraza pojavita skupaj, če sta statistično neodvisna. Razmerje med tema verjetnostma izraža nivo statistične odvisnosti med izrazoma. Verjetnosti so izračunane s pomočjo štetja števila zadetkov na enem izmed spletnih iskalnikov (Turney je v [25] uporabil takrat obstoječi iskalnik

	Prva beseda	Druga beseda	Tretja beseda (ni izluščena)
1.	pridevnik	samostalnik	karkoli
2.	prislov	pridevnik	ni samostalnik
3.	pridevnik	pridevnik	ni samostalnik
4.	samostalnik	pridevnik	ni samostalnik
5.	prislov	glagol	karkoli

Tabela 2.1: Vse iskane oblikoslovne oznake dveh izluščenih besed, kot jih je določil Turney [25].

AltaVista). Za vsak iskani izraz vrne iskalnik število relevantnih zadetkov. Z iskanjem števila zadetkov za $fraza$ in $referenca$ skupaj in za vsak izraz posebej lahko ocenimo verjetnosti v enačbi (2.2).

Korak 3: Pri danem mnenjskem dokumentu izračunamo povprečje mnenjskih orientacij vseh najdenih fraz. Če je izračunano povprečje pozitivno, potem dokument razvrstimo v pozitiven razred, v nasprotnem primeru pa v negativen razred.

Pristopi, ki so sorodni opisanemu, temeljijo na uporabi leksikonov, ki uporabljajo slovar mnenjskih besed in fraz z njihovimi pripadajočimi orientacijami. Generacijo leksikonov za uporabo pri analizi sentimenta bomo podrobneje opisali v razdelku 2.6.

2.4 Nivo povedi

Včasih klasifikacija na nivoju dokumenta ni ustrezna, ker lahko mnenjsko besedilo izraža različna mnenja o različnih atributih objekta. Analiza sentimenta na nivoju povedi je razdeljena na dva dela. V prvem delu ugotavljamo subjektivnost povedi, v drugem pa mnenjsko orientacijo subjektivne povedi. Tako lahko ti dve nalogi razdelimo na:

Označevanje subjektivnosti: v tem koraku označimo povedi v mnenjskem besedilu kot subjektivne ali objektivne. Objektivne povedi iz-

ločimo na predpostavki, da ne vsebujejo nobenega mnenja.

Označevanje sentimenta na nivoju povedi: ko smo določili vse subjektivne povedi, jih v tem koraku označimo kot pozitivne ali negativne.

V spodnjem primeru (B), bi prvo poved klasificirali kot objektivno, ostali dve pa kot subjektivni. Nato bi iz subjektivnih povedi izluščili sentiment, ki bi bil v prvem primeru pozitiven in v drugem negativen. Za oba koraka lahko uporabimo že znane metode nadzorovanega učenja in statistične analize, ki so bile opisane v prejšnjih poglavjih.

"I bought a new Nexus. I really like how responsive device is. However, picture quality is quite poor." (B)

Ker je analiza sentimenta na nivoju povedi ponavadi samo vmesni korak, pri njej ne definiramo petorke (o, f, s, h, t). Prav tako na tem nivoju predpostavljamo, da posamezna subjektivna poved izraža enojno mnenje znanega izražalca mnenja o znanem objektu ali njegovem atributu. Vendar to zmeraj ne drži, ker obstajajo povedi kot so *"Actors in this movie were great but the whole story was extremely boring."*, ki izražajo več mnenj o različnih atributih objekta. Prav tako z analizo sentimenta na tem nivoju ne moremo dobro obravnavati primerjalnih povedi, kot so *"I like iPhone but Nexus is just better."*. Nenazadnje ne smemo pozabiti, da lahko tudi objektivne povedi izražajo sentiment.

2.5 Nivo entitete in aspekta

Kot smo opisali na koncu prejšnjega razdelka, tudi analiza sentimenta na nivoju povedi ne obravnava vseh primerov, kjer oseba izrazi svoje mnenje o določenem objektu in njegovih atributih. To postane še posebej problematično v daljših mnenjskih dokumentih, kot so blogi, kjer lahko izražalec mnenja opisuje nek produkt in storitev in izraža pozitivno in negativno mnenje o različnih lastnostih le-tega. Poleg tega lahko mnenjski dokument izraža

mnenja več oseb, kot smo že predhodno omenili. V tem primeru moramo narediti analizo sentimenta, ki sledi vsem korakom opisanim v poglavju 2.2. Tako moramo iz mnenjskega dokumenta izluščiti vse petorke (o, f, s, h, t), da dobimo natančen pregled nad tem, kaj je bilo v besedilu opisano, kakšna so bila mnenja, o čem so bila podana in kdo jih je podal.

Določanje izražalca mnenja, objekta in časovne komponente spada v področje luščenja informacij (*Named Entity Recognition*) in ga tu podrobneje ne bomo obravnavali. Ekstrakcija atributov objekta je trenutno zelo raziskovano področje, ki obravnava različne pristope glede na strukturo mnenjskega dokumenta in so podrobneje predstavljeni v [16,18]. Za določanje orientacije mnenja posamezne petorke lahko uporabimo vse metode, ki so uporabljene za določanje sentimenta na nivoju povedi. V literaturi [23] so raziskane tudi druge, ki so bolj kompleksne in jih tu ne obravnavamo.

2.6 Leksikoni mnenjskih besed

Leksikoni mnenjskih besed so zbirke mnenjskih besed in fraz s pripadajočimi mnenjskimi orientacijami in se uporabljajo pri pristopih za določanje sentimenta na vseh nivojih analize dokumenta. Za gradnjo zbirke mnenjskih besed lahko uporabimo tri pristope: ročnega, osnovanega na slovarjih in osnovanega na besedilnem korpusu. Ročni pristop je časovno zelo potraten in je zato najpogosteje uporabljen v kombinaciji z drugimi pristopi.

2.6.1 Leksikoni, osnovani na besednih slovarjih

Mnenjski leksikoni, osnovani na slovarjih, so zgenerirani s pomočjo manjše referenčne množice mnenjskih besed in obstoječega slovarja besed. Za izbrane besede že vnaprej določimo orientacijo. Nato v prvi iteraciji pogledamo v besedni slovar vsako od besed in poiščemo vse njene sinonime in protipomenke. Nato te novo-najdene besede dodamo v referenčno množico in postopek ponovimo. Proces gradnje leksikona se konča, ko v iteraciji ne najdemo več novih besed. Nato ročno pregledamo leksikon in popravimo morebitne napačno

	MPQA	Opinion Lexicon	Inquirer	SentiWordNet	LIWC
MPQA	-	0.6%	2%	27%	3%
Opinion Lexicon		-	1%	25%	2%
Inquirer			-	23%	0.5%
SentiWordNet				-	25%
LIWC					-

Tabela 2.2: Tabela prikazuje stopnje nestrinjanja glede orientacije besed. Vir: http://sentiment.christopherpotts.net/lexicons.html

razvrščene besede. Naprednejše metode lahko uporabijo tudi nadzorovano učenje. Glavne pomanjkljivosti tega pristopa so, da leksikon mogoče ni dobro prilagojen za določeno problemsko domeno, kjer imajo nekatere mnenjske besede nasprotno orientacijo tisti, ki je določena v leksikonu.

2.6.2 Leksikoni, osnovani na besedilnem korpusu

Težave leksikonov, ki so osnovani izključno na besednih slovarjih, lahko rešimo z metodami, ki temeljijo na sintaktičnih pravilih ali pravilih sopojavljanja besed. Ena od tehnik, ki je bila predstavljena v [9], začne z referenčno množico pridevnikov, ki išče ostale pridevnike v besedilnem korpusu. Pri tem upošteva lingvistična pravila. Če stavek, ki ga obravnavamo, vsebuje dva pridevnika ločena z vezno besedo “in”, kjer je eden izmed njih naša referenčna beseda, potem drugemu pripišemo enako orientacijo. Podobna pravila obstajajo tudi za druge vezne besede (“ali”, “vendar”), ki novo najdenim pridevnikom pripišejo primerno orientacijo glede na referenčni pridevnik in vezno besedo.

Veliko leksikonov je že zgeneriranih na različnih korpusih, njihovo primerjavo pa lahko vidimo v tabeli 2.2, ki prikazuje odstotek besed, katerim so leksikoni pripisali nasprotne orientacije. Glavna pomanjkljivost teh leksikonov je, da so zgenerirani na besedilnem korpusu neke domene in če korpus ni dovolj velik, je velika verjetnost, da ne zajame vseh besed nekega jezika. Po drugi strani pa so korpusi zelo domensko specifični in imajo besede v njem pravilno orientacijo glede na področje, ki ga obravnavajo.

2.7 Orodja za analizo sentimenta

Analiza sentimenta je postala zelo zanimivo področje tako za raziskovanje kot tudi za praktično uporabo. Zaradi tega obstaja poleg orodij tudi veliko podjetij, ki se ukvarjajo z razvojem in nudenjem storitev na tem področju. V tem poglavju bomo na kratko opisali nekaj orodij in storitev. Nekaj izmed njih bomo v zadnjem poglavju primerjali z našim sistemom.

2.7.1 VADER

Eno takih orodij je VADER (Valence Aware Dictionary and sEntiment Reasoner) [12], ki temelji na uporabi leksikona in modela pravil in je posebej optimiziran za področje mnenjskih besedil, ki se pojavljajo na družabnih omrežjih. Metode in procesi, opisani v [12], zajemajo naslednje korake:

1. Razvoj in validacija leksikona mnenjskih besed, ki imajo negativno ali pozitivno polarnost in intenziteto med -4 in +4. Leksikon je bil razvit in ovrednoten ročno s pomočjo obstoječih leksikonov in dodajanjem novih domensko specifičnih izrazov.

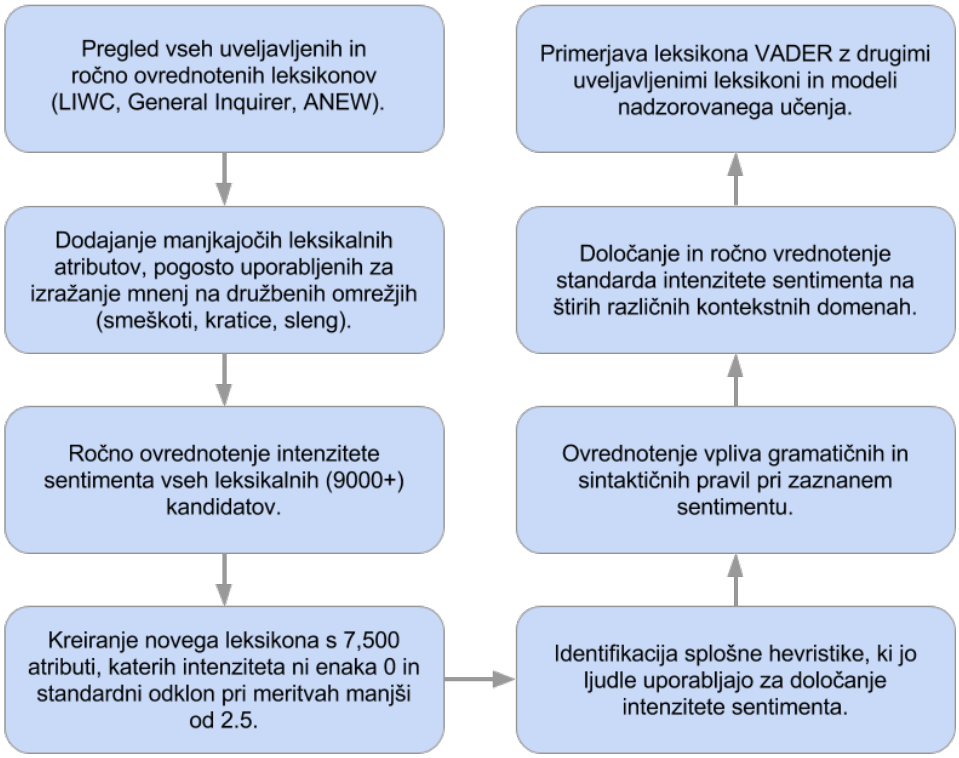
2. Identifikacija in ocenitev modela pravil za konvencionalno uporabo pri gramatičnih in sintaktičnih aspektih besedila za določanje polarnosti.

3. Ocenitev sistema in primerjava z ostalimi ustaljenimi metodami za analizo sentimenta.

Bolj podrobna struktura metod in procesov je prikazana na sliki 2.1.

VADER je bil tudi implementiran in je prosto dostopen na spletu v obliki paketa za Python[3]. Zajema gramatična in sintaktična pravila, opisana v članku, in vsebuje empirično pridobljene vrednosti, ki določajo vpliv posameznega pravila pri izračunu orientacije mnenja.

[3]https://pypi.python.org/pypi/vaderSentiment



Slika 2.1: Metode in procesi uporabljeni pri snovanju sistema VADER.

2.7.2 Komercialne storitve

Poleg zgoraj opisanega sistema, ki temelji na raziskovalnem članku, pa obstaja tudi ogromno podjetij, ki ponujajo strojno učenje ali analizo sentimenta kot storitev. Skupno jim je to, da skrijejo večino podrobnosti in naredijo storitev uporabno brez predhodnega poznavanja področja strojnega učenja ali analize sentimenta.

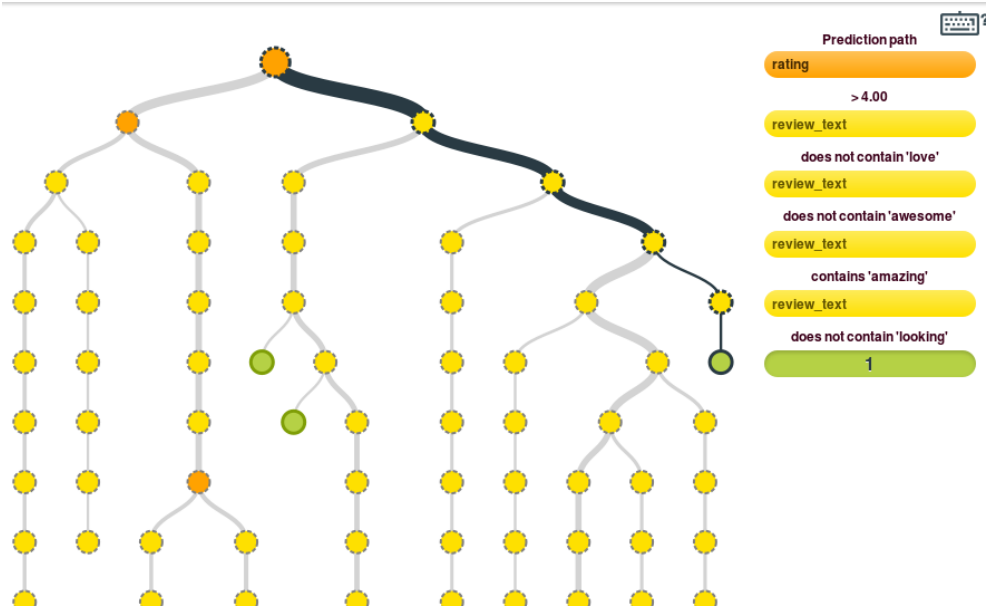
Indico: podjetje ponuja različne storitve za analizo besedil in slik na podlagi strojnega učenja. Za analizo sentimenta v besedilih ponuja dve različni metodi nadzorovanega učenja[4]. Prva metoda zajema izračun TF-IDF (anlg. *term frequency–inverse document frequency*), izbor n-gramov in logistično regresijo za klasifikacijo. Druga metoda pa temelji na rekurenčnih nevronskih mrežah (*recurrent neural networks*).

[4]https://indico.io/news/121515_indico_SentimentHQ_Release

Google Prediction API: spada v sklop storitev Google Cloud Platform in ponuja sklop storitev za analizo besedil s pomočjo strojnega učenja. O podrobnostih sistema za analizo sentimenta ni veliko podatkov, ker gre za lastniški in zaprt sistem.

Amazon ML: je storitev, ki omogoča preprosto gradnjo modelov z nadzorovanim učenjem in spada med storitve Amazon Web Services.

BigML: je storitev MlaaS (angl. *machine learning as a service*) in ponuja izjemno preprosto grajenje odločitvenih dreves in njihovo vrednotenje, kot prikazuje primer na sliki 2.2



Slika 2.2: Prikaz odločitvenega drevesa pri uporabi spletne storitve BigML. Na sliki lahko vidimo zgrajeno odločitveno drevo in prikaz pravil ene od odločitvenih vej.

Poglavje 3

Analiza vhodnih podatkov

Naša raziskovana domena so komentarji, ki so v spletni trgovini Google Play pod aplikacijami za mobilne telefone. Poznavanje strukture komentarjev je ključno za izbor pravilnih metod, ki jih bomo uporabili pri gradnji našega sistema, zato si jih bomo v tem poglavju ogledali podrobneje. Zbirko komentarjev smo pridobili v strukturirani obliki in je last podjetja AppMonsta. Vsak komentar je shranjen v obliki dokumenta JSON, kot je prikazano na sliki 3.1. Zbirka je zelo obsežna in vsebuje prek 230 milijonov angleških komentarjev, od tega jih je 8718 ročno označenih. Označeni so bili s pomočjo več ocenjevalcev in vsak komentar je bil označen vsaj dvakrat. Ročno označeni komentarji imajo v strukturi dokumenta dodatna polja, ki označujejo vrednosti binarnega razreda, podane s strani ocenjevalcev. Ocenjeni so na podlagi tega, ali izražajo navdušenje ali ne. Na primer, komentar na sliki 3.1 bi bil označen kot pozitiven. Vsak komentar vsebuje dve polji: *"title"* in *"review_text"*, kjer se nahaja vsebina komentarja.

3.1 Dolžina komentarjev

Z analizo dolžine komentarjev želimo dobiti boljši vpogled v to, s kakšnimi besedili imamo opravka. V tabeli 3.1 lahko vidimo, da je velika večina komentarjev enopovednih ali dvopovednih. Opazimo lahko, da so pozitivni

Slika 3.1: Prikaz komentarja prisotnega na trgovini Google Play in v strukturirani JSON obliki.

komentarji bolj pogosto izraženi z eno povedjo, kot pa tisti, ki navdušenja ne izražajo. Prav tako vsi komentarji nimajo izpolnjenega polja *"title"*. Takih je 36% med komentarji brez navdušenja, in 29% med tistimi, ki izražajo navdušenje.

3.2 Sestava besed v komentarjih

Komentarje smo analizirali glede na število besed, ki se pojavljajo v navdušujočih in ostalih komentarjih. Iz analize smo izločili zelo pogoste besede in besedne fraze, ki spadajo v zbirko členkov, veznikov, zaimkov in drugih kratkih besed (angl. *stop-words*). Med njimi najdemo besede, kot so "the", "that", "I" in podobne. Te besede so zelo pogoste v vseh besedilih in jih ponavadi pred besedilno analizo izločimo. V tabeli 3.2 lahko vidimo, da so v komentarjih z navdušenjem najbolj pogosto uporabljeni pozitivni pridevniki in glagoli. Zanimivo je, da tudi v komentarjih brez navdušenja prevladujejo pozitivne besede. Eden od razlogov je ta, da iščemo komentarje, ki izražajo navdušenje in ne samo pozitivno mnenje. Poleg tega pa je lahko z negacijo pozitivnim besedam v komentarjih obrnjena orientacija.

	Vsi komentarji	Komentarji z navdušenjem	Brez navdušenja
1 poved	76.9%	80.9 %	74.3%
2 povedi	11.2%	10.6%	11.6%
3 povedi	4.8%	4.4%	5.1%
4 povedi	2.2%	1.6%	2.5%

Tabela 3.1: Primerjava števila povedi v polju *"review_ text"* v vseh označenih komentarjih. Prvi stolpec prikazuje odstotke komentarjev, ki so zgrajeni iz ene, dveh, treh ali štirih povedi. Druga dva stolpca prikazujeta odstotke, dobljene samo na komentarjih z navdušenjem, in tistih, ki navdušenja ne izražajo.

3.3 Pravilnost črkovanja in nebesedni znaki

Komentarji v naši zbirki so izrazito podvrženi slengovskemu izražanju in kar 36% besed je napačno zapisanih. V to so sicer všteti tudi samostalniki, ki opisujejo lastna imena, kot je na primer "Facebook", ki jih sistem za preverjanje črkovanja ni prepoznal. V tabeli 3.3 lahko vidimo, da so večina teh besed nepravilno zapisani samostalniki, pridevniki in ostale besedne vrste. V komentarjih poleg besednih vrst nastopajo tudi nebesedni simboli, kot so: ☺ 😎 😄 😕 ☹ in podobni. Ti simboli predstavljajo razpoloženje ali čustvo in so prisotni v 4% naših komentarjev.

3.4 Porazdelitev komentarjev glede na oceno

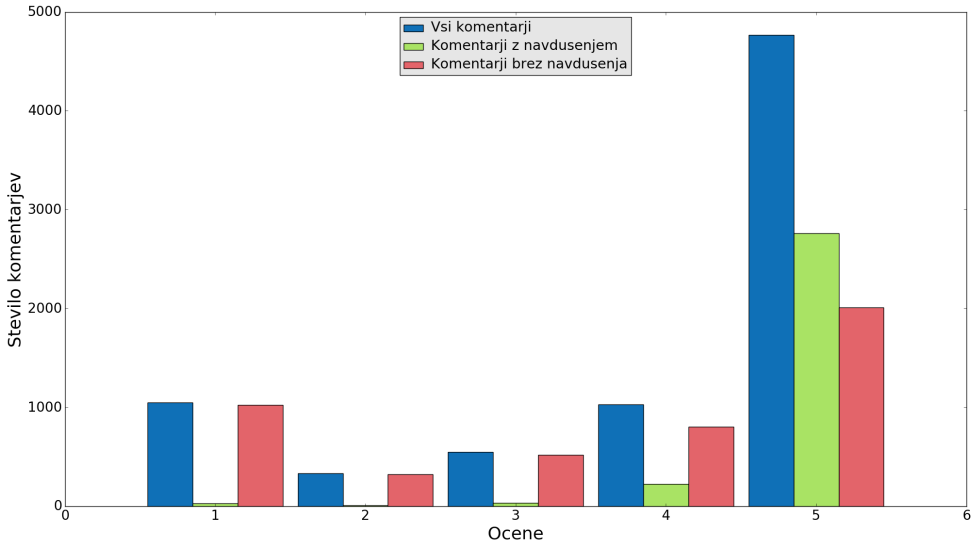
Vsak komentar ima tudi pripadajočo oceno, ki je bila podana s strani avtorja komentarja. Če pogledamo porazdelitev komentarjev glede na oceno in ročno oznako, lahko na sliki 3.2 vidimo, da je večina komentarjev ocenjenih z oceno 5 in da je velika večina komentarjev, ki izražajo navdušenje, v tem razredu. Zanimivo je, da polovica komentarjev z oceno 5 ne izraža navdušenja.

	Komentarji z navdušenjem	Brez navdušenja
1	love	good
2	game	app
3	app	game
4	awesome	nice
5	best	like
6	great	great
7	good	time
8	fun	fun
9	amazing	get
10	loved	use
11	really	play
12	ever	please
13	like	update
14	easy	fix
15	cool	really
16	play	one
17	use	work
18	one	would
19	nice	even
20	time	easy

Tabela 3.2: Prikazanih je 20 najpogosteje uporabljenih besed v komentarjih z in brez navdušenja.

	Beseda	Število pojavitev
1	awsome	83
2	dont	73
3	plz	58
4	lol	51
5	im	41
6	pls	34
7	alot	30
8	omg	26
9	fb	26
10	wtf	23
11	coc	18
12	everytime	18
13	gameplay	16
14	ui	16
15	gud	15
16	awsm	14
17	aap	14
18	luv	14
19	doesnt	12
20	usefull	12

Tabela 3.3: 20 najbolj pogosto napačno zapisanih besed. Število pojavitev predstavlja kolikokrat se posamezna beseda pojavi v naših komentarjih.



Slika 3.2: Porazdelitev komentarjev glede na oceno. Vidimo lahko, da je večina komentarjev z označena z oceno 5. Od teh jih več kot polovica predstavlja tiste, ki izražajo navdušenje. Prav tako lahko opazimo, da komentarjev z navdušenjem, ki imajo oceno 1, 2 ali 3, skoraj ni. V veliki večini so ocenjeni z oceno 5.

3.5 Izbor komentarjev

Ker so bili vsi komentarji označeni večkrat, izločimo iz naše učne množice tiste, pri katerih sta ocenjevalca dala komentarju različno oznako razreda. Takih komentarjev je 12%. Ker ne želimo delati nobenih predpostavk glede reprezentativnosti razredov, bomo v naši učni množici uporabili enako zastopanost obeh. S tem je velikost vhodne množice, ki jo uporabimo pri učenju in vrednotenju, omejena na 5470 komentarjev.

Poglavje 4

Analiza sentimenta s pomočjo nadzorovanega učenja

V tem poglavju bomo zgradili sistem, ki bo na podlagi prej opisanih vhodnih podatkov razvrstil komentarje na tiste, ki izražajo navdušenje, in tiste, ki ga ne. Uporabili bomo metode nadzorovanega učenja in klasifikacijo na nivoju dokumenta, ki so opisane v poglavju 2.3. Naši vhodni podatki bodo mnenjski dokumenti, ki so predstavljeni v obliki komentarjev, in vsak komentar d vsebuje vnaprej znana polja:

1. Objekt o, na katerega se navezujejo mnenja, je znan. To bo aplikacija, pod katero naš komentar spada in je prisotna v dokumentu v polju "app_id".

2. Izražalec mnenja h, ki je avtor komentarja, je prav tako znan in je določen v dokumentu v polju "user_id".

3. Časovna komponenta t je znana in predstavlja datum nastanka komentarja ter je podana v dokumentu v polju "date".

Pri tem predpostavljamo, da izražalec mnenja izraža splošno mnenje o samem objektu in ne o posameznih atributih objekta. Tako bo naša naloga določiti sentiment s za vsak vhodni dokument, da dobimo petorko

Vhodno besedilo	Žetoni
I like this car.	I, like, this, car
This is AWESOME!!!!!	This, is, AWESOME!!!!!

Tabela 4.1: Primer razčlenjevanja besedila na praznih znakih.

$(o, SPLOSNO, s, h, t)$. Pri tem smo v petorki vnaprej določili atribut $SPLOSNO$, ki je rezultat naše predpostavke, da je mnenje podano o samem objektu in ne o različnih atributih objekta.

4.1 Razčlenjevanje

Razčlenjevanje je proces, kjer besedilo razbijemo v žetone besed, števil, fraz in simbolov. Je prvi korak pri obdelavi vhodnega besedila v naravnem jeziku za kasnejšo analizo. Pri tem obstaja veliko načinov, kako to narediti, in najbolj primerna metoda se določi glede na vrsto analize ter vrsto besedila, s katero imamo opravka. Najbolj preprost način razčlenjevanja je, da vhodno besedilo razčlenimo po presledkih, novih vrsticah in ostalih praznih znakih (angl. *whitespace characters*). Primer te metode je prikazan v tabeli 4.1.

Ponavadi tako preprosto razčlenjevanje ni dovolj dobro, ker ostanejo posamezne besede in ločila v istem žetonu. Primer tega sta žetona “amazing,” in “amazing.”, ki vsebujeta isto besedo, ampak bosta zaradi drugačnega ločila na koncu, obravnavana kot različni besedi. Ponavadi tega ne želimo in moramo zato izbrati pravo metodo, kjer so dobljeni žetoni lingvistično pomembni in metodološko uporabni pri naslednjih korakih. Napake, narejene v tem koraku, se prenesejo v vse sledeče korake pri analizi teksta. Zaradi tega se razčlenjevanje vhodnega besedila razlikuje glede na domeno problema in smo jo pri našem problemu razdelili na naslednje korake:

1. Iskanje vseh besed in števil v besedilu.
2. Iskanje vseh nebesednih znakov, ki izražajo čustva.
3. Iskanje vseh ostalih nepraznih znakov, kot so ločila in simboli.

Vrsta	Regularni izraz	Ujemanje
Besede	`[a-z][a-z'\-]+[a-z]|[\w]+}`	love, don't, AWESOME, beeeest.
Števila	`\d+[,.]?\d*`	5, 10, 3.0
Smeškoti	`[<>]?[:;=8][\-o\*\']?[\)\]` `\(\[dDpP/\:\}\{@|\\<>]`	:), :-(, >:P
Simboli in ločila	`\S+`	vsi ostali neprazni znaki, npr. ločila in Unicode simboli

Tabela 4.2: Sklop regularnih izrazov, ki jih uporabimo pri razčlenjevanju in primeri dobljenih žetonov.

Pri gradnji našega sistema za razčlenjevanje si pomagamo z regularnimi izrazi in za vsako zgornjo točko sestavimo regularni izraz, ki nam služi za iskanje želenega žetona v danem vhodnem besedilu. Ker večina ljudi danes izraža čustva s pomočjo posebnih znakov, smo pozorni, da zajamemo tudi spekter Unicode simbolov[1], ki izražajo čustva, in ne samo znakov, ki so del sklopa ASCII. V tabeli 4.2 lahko vidimo uporabljene regularne izraze in primere dobljenih žetonov.

4.2 Normalizacija in negacija

Naslednji korak, ki ga naredimo, je normalizacija besed in znakov [4]. Velikokrat ljudje izražajo čustva skozi način pisanja, kot so uporaba velikih črk in podaljševanje besed. Tako ponavadi *"This is good"* in *"This is goooooooooooood"* izražata drugačen nivo čustev in vznemirjenja. Prav tako *"THIS IS GOOD"* doda intenziteto napisani frazi. Zato bomo v naslednjem koraku poiskali vse besede, kjer je neka črka zaporedno ponovljena več kot dvakrat. Nato bomo vse te besede normalizirali. Enak pristop bomo privzeli tudi pri nebesednih sklopih znakov, ker tudi v primeru *"This is good!!!!!!!!!"*, avtor s klicaji potencira izraženo čustvo. Normalizacija zajema sledeče korake:

[1] http://www.unicode.org/Public/emoji/3.0/

Korak 1: iskanje vseh žetonov, pri katerih se nek znak zaporedno ponovi več kot dvakrat.

Korak 2: vsak tak žeton potem normaliziramo, kjer število ponovljenih znakov zmanjšamo na tri. Tako se *"This is goooooood!!!!"* spremeni v *"This is goood!!!"*. Izjema so le Unicode simboli, kjer število ponovljenih znakov zmanjšamo na ena.

Število ponovljenih znakov tako normaliziramo na tri (izjema so Unicode znaki), ker želimo ohraniti informacijo o intenziteti in razlikovati med pravilno zapisanimi besedami in tistimi, ki so bile podaljšane v originalnem zapisu. Velikokrat se pri analizi besedil dela tudi normalizacija, kjer vhodne besede spremenimo v besede, zapisane z malimi črkami, in to metodo bomo uporabili tudi mi. Ker s tem ne želimo izgubiti informacije o intenziteti besedila, ki se odraža z veliko pisavo, to informacijo predhodno shranimo.

Na koncu opravimo še proces negacije [4], kjer želimo označiti besede, ki dobijo nasprotno polarnost zaradi uporabe negativnih besed. Negativne besedne vrste, ki obrnejo polarnost, so prikazane v tabeli 4.3.

Negativne besede	no, not, none, nobody, nothing, neither, nowhere, never
Negativni pridevniki	hardly, scarcely, barely
Negativni glagoli	vse besede, ki se končajo z n't (doesn't, isn't, wasn't, ...)

Tabela 4.3: Negativne besedne vrste, ki obrnejo polarnost.

Negacijo opravimo z naslednjimi koraki:

Korak 1: V vsakem razčlenjenem dokumentu poiščemo žeton, ki ustreza negativni besedi.

Korak 2: Temu žetonu in vsem sledečim žetonom pripnemo oznako $\_NEG$ na koncu niza. S pripenjanjem oznake končamo, ko najdemo žeton, ki

označuje konec povedi. Žetoni, ki ozačujejo konec povedi, so izraženi kot eno ali več ločil ([. :; !?]+). Primer negacije je prikazan v tabeli 4.4.

Vhodni niz pred razčlenjevanjem	not very good. keeps crashing
Niz žetonov pred negacijo	not, very, good, ., keeps, crashing
Niz žetonov po negaciji	not_NEG, very_NEG, good_NEG, ., keeps, crashing

Tabela 4.4: Primer negacije niza žetonov. Najprej vhodni niz razčlenimo v niz žetonov, ki ga nato negiramo. Dobljeni žetoni so ločeni z vejico.

4.3 N-grami

Naslednji korak pri pripravi vhodnega besedila v vektorsko obliko primerno za klasifikacijo je transformacija vseh razčlenjenih nizov v vreče besed (angl. *bag of words*). Pri tem izgubimo vrstni red žetonov v vhodnem nizu ter s tem nekatere slovnične lastnosti. Da minimiziramo izgubo teh informacij, pred tem ustvarimo n-grame iz naših žetonov. N-grami so definirani kot sekvence zaporednih besed v vhodnem nizu in s tem lahko ohranimo nekatere fraze, ki bi se drugače izgubile v vreči besed. Na primer v primeru niza "Screen is very bad and battery life not good.", se izgubijo fraze "not good" in "very bad". Določen del informacije se sicer ohrani skozi proces negacije, vendar da ohranimo tudi ostale fraze, zgradimo n-grame iz niza vhodnih žetonov. Najbolj preprosti n-grami so 1-grami, ki zajemajo eno besedo in naši žetoni so že v tej obliki. Naslednja stopnja so 2-grami in 3-grami, ki skupaj povežejo sosednje žetone, kot je prikazano v tabeli 4.5.

Poleg n-gramov na nivoju vhodnega niza poznamo tudi n-grame na nivoju besede. Pri tem generiramo n-grame na vsakem vhodnem žetonu. Tako zgenerirani n-grami sicer v večini primerov ne predstavljajo več veljavnih besed nekega besednjaka, vendar nam lahko pomagajo pri besedilih, ki imajo

1-grami	screen, is, very, bad
2-grami	screen is, is very, very bad
3-grami	screen is very, is very bad

Tabela 4.5: Primeri različnih n-gramov na razčlenjenem nizu.

veliko slovničnih napak. V našem korpusu komentarjev je na primer velikokrat napačno zapisana beseda "awesome" kot "awsome". V tabeli 4.6 lahko vidimo, da če iz obeh besed ustvarimo 2-grame in le-te uporabimo v vreči besed, bomo dosegli večjo pokritost med njima. Pri 1-gramskih vrednostih bosta imela končna vektorja dva popolnoma različna atributa, pri 2-gramskih vrednostih pa bosta imela štiri od šestih atributov enakih.

awesome	aw, we, es, so, om, me
awsome	aw, ws, so, om, me

Tabela 4.6: Primer 2-gramov ustvarjen na besedi "awesome" in "awsome"

4.4 Frekvenca pojavitev žetona

Po generiranju n-gramov iz vhodnih razčlenjenih nizov je naslednji korak generiranje vreče besed za vsak vhodni niz. Ta reprezentacija je kasneje primerna za pretvorbo podatkov v atributni zapis, kar na koncu pričakuje izbrani klasifikator. V posamezni vreči besed so predstavljeni žetoni, ki bodo predstavljali atribute, vrednost atributov pa določimo z različnimi metodami. Pri klasifikaciji besedilnih dokumentov se največkrat uporabljajo prisotnost žetona, število pojavitev žetona v besedilu, frekvenca pojavitev žetona v besedilu in mera TF-IDF. Frekvenca pojavitev žetona v vhodnem nizu je razdeljena na več različno uteženih metod [5], in sicer:

Binarna: definiramo jo kot funkcijo, ki prejme žeton in vhodni dokument in vrne 1, če je žeton prisoten v dokumentu in 0, če ni.

Normalna: je funkcija, ki prejme žeton in vhodni dokument in vrne število pojavitev žetona v vhodnem dokumentu [2] in jo izračunamo z enačbo (4.1):

$$\begin{aligned} tf(t,d) &= \sum_{i \in d} eq(i,t) \\ eq(i,t) &= \begin{cases} 1, & i = t; \\ 0, & \text{sicer} \end{cases} \end{aligned} \tag{4.1}$$

Normalizirana: ta funkcija uporablja logaritemski način uteževanja in jo izračunamo s pomočjo enačbe:

$$tf_{log}(t,d) = 1 + log(tf(t,d)) \tag{4.2}$$

Poleg zgoraj predstavljene normalizirane metode obstajajo še druge, ki vrednost pojavitve normalizirajo tudi na podlagi dolžine vhodnega besedila. Poleg frekvence pojavitev je na področju pridobivanja informacij (angl. *information retrieval*) zelo uporabljena tudi mera TF-IDF. Pri prej opisanih metodah je problem to, da žetoni, ki se pojavljajo pogosto pri velikem deležu vhodnih dokumentov, velikokrat spadajo pod splošne besede (npr. veznike), ki pri reševanju problema ne nudijo informacije in lahko tudi zamaskirajo žetone, ki so dejansko pomembni. Zato pri TF-IDF računamo frekvenco pojavitve nekega žetona v vhodnem besedilu in frekvenco pojavitve tega žetona v vseh vhodnih besedilih. Na ta način so kot pomembni žetoni označeni tisti, ki imajo veliko frekvenco pojavljanja v trenutnem besedilu in hkrati majhno v celotnem korpusu. Pri tem za vsak žeton t najprej izračunamo df_t, ki predstavlja število vhodnih dokumentov v celotnem korpusu, kjer se pojavlja t. Število vseh vhodnih dokumentov označimo z N. Nato izračunamo $idf(t)$ s pomočjo formule (4.3) in na koncu $tfidf(t)$, kot je prikazano v formuli (4.4).

$$idf(t,d) = log(N/df_t) \tag{4.3}$$

$$tfidf(t,d) = idf(t,d) \cdot tf_{log}(t,d) \tag{4.4}$$

4.5 Korenjenje

Korenjenje (angl. *stemming*) je proces, kjer odstranimo zadnji del besede in ohranimo njen koren. S tem želimo zmanjšati velikost besednjaka in poenotiti različne izpeljanke besed, ki imajo v osnovi enak pomenski izvor [4]. Za angleški jezik je eden najbolj uporabljenih in empirično najboljših algoritmov Porterjev algoritem (angl. *Porter's algorithm*) [20]. Algoritem uporablja hevristiko in sistem pravil, ki kar najbolje odrežejo končnice besed. Sestavljen je iz petih korakov[2] in v tabeli 4.7 je prikazan prvi korak. Pri uporabi korenjenja pri analizi sentimenta moramo biti previdni, ker se lahko orientacija sentimenta pri tem izgubi [4]. V tabeli 4.8 so prikazani pari besed iz leksikona Harvard General Inquirer[3] z obratno orientacijo a enakim korenom.

Pravilo	Primer
$SSES \rightarrow SS$	losses $\rightarrow$ loss
$IES \rightarrow I$	ponies $\rightarrow$ poni
$SS \rightarrow SS$	glass $\rightarrow$ glass
$S \rightarrow \emptyset$	dogs $\rightarrow$ dog

Tabela 4.7: Prvi sklop pravil pri uporabi Porterjevega algoritma za korenjenje besed.

Pozitivna beseda	Negativna beseda	Koren
defense	defensive	defens
tolerance	tolerable	toler
dominance	dominate	domin
objective	objection	object

Tabela 4.8: Primeri pozitivnih in negativnih besed iz leksikona Harvard General Inquirer, ki dobijo pri korenjenju enak koren. Pri tem se izgubi orientacija sentimenta.

Zaradi tega uporaba korenjenja pri analizi sentimenta ni samoumevna in

[2]http://snowball.tartarus.org/algorithms/porter/stemmer.html
[3]http://www.wjh.harvard.edu/ inquirer/

lahko rezultate klasifikacije izboljša ali pa poslabša. Ali je uporaba upravičena, je odvisno od domene problema in empiričnih rezultatov.

4.6 Popravljanje črkovanja

Za popravljanje črkovanja bomo uporabili model kanala s šumom (angl. *noisy channel model*) [8,15] pri katerem je cilj najti pravilno besedo w v besednjaku V pri podani nepravilno zapisani besedi x. Iskano besedo določimo tako, da generiramo nabor kandidatov in izberemo tistega z največjo verjetnostjo. Metoda uporablja Bayesovo pravilo in jo zapišemo kot:

$$\hat{w} = \underset{w \in V}{\operatorname{argmax}} P(w|x) \tag{4.5}$$

$$= \underset{w \in V}{\operatorname{argmax}} \frac{P(x|w)}{P(w)} P(x) \tag{4.6}$$

$$= \underset{w \in V}{\operatorname{argmax}} P(x|w)P(w) \tag{4.7}$$

Pri tem imenujemo $P(x|w)$ model napake (angl. *error model*) in $P(w)$ jezikovni model (angl. *language model*). Model napake izračuna verjetnost, da pri pisanju besede w zapišemo napačno besedo x. Jezikovni model pa oceni verjetnost pojavitve besede w v danem kontekstu. Pri tej metodi najprej izračunamo možne kandidate w s pomočjo Daerau-Levenshteinove razdalje urejanja (angl. *Damerau-Levenshtein edit distance*). Razdalja predstavlja minimalno število sprememb v besedi x, da iz nje dobimo w. Veljavne spremembe nad besedo so: vstavljanje nove črke, brisanje obstoječe črke, zamenjava obstoječe črke z novo in zamenjava dveh sosednjih črk. Ker so v 80% primerih napake narejene na razdalji 1[4], bomo naše popravljanje omejili na to dolžino. To pomeni, da bomo predpostavili, da potrebujemo natanko eno spremembo, da dobimo iz nepravilno zapisane besede, pravilno. Verjetnost modela napake za vsakega kandidata w izračunamo kot verjetnost, da

[4]https://web.stanford.edu/class/cs124/lec/spelling.pdf

je prišlo do točno določene spremembe, ki je besedo w spremenila v x. Verjetnosti so izračunane s pomočjo štirih matrik: $del[x, y]$, $ins[x, y]$, $sub[x, y]$ in $trans[x, y]$. Te matrike vsebujejo pogostosti različnih napak v učni množici. Tako nam $del[x, y]$ pove, kako pogosto je bila črka y izbrisana za črko x. To vrednost nato normaliziramo s številom pojavitev xy v učni množici. Celotno enačbo za izračun verjetnosti modela napake tako zapišemo kot:

$$P(x|w) = \begin{cases} \frac{del[w_{i-1},w_i]}{count[w_{i-1}w_i]} & pri\ brisanju \\ \frac{ins[w_{i-1},x_i]}{count[w_{i-1}]} & pri\ vstavljanju \\ \frac{sub[x_i,w_i]}{count[w_i]} & pri\ zamenjavi \\ \frac{trans[w_i,w_{i+1}]}{count[w_iw_{i+1}]} & pri\ zamenjavi\ sosednjih\ crk \end{cases} \tag{4.8}$$

Pri tem $count[x]$ predstavlja število pojavitev niza x v učni množici. S pomočjo jezikovnega modela izračunamo verjetnost pojavitve nekega niza besed. Pri tem smo se odločili testirati Laplaceov 1-gramski jezikovni model (angl. *Laplace unigram language model*), Laplaceov 2-gramski jezikovni model (angl. *Laplace bigram language model*) in model neumnega sestopanja (angl. *stupid backoff*) [7]. Laplaceov n-gramski jezikovni model je pri danem zaporedju besed $w_1^n = w_1...w_n$ definiran kot:

$$P(w_1^n) = \prod_{k=1}^{n} P(w_k|w_{k-N+1}^{k-1}) \tag{4.9}$$

Pri tem pogojne verjetnosti izračunamo s pomočjo relativne frekvence niza besed v učni množici in pri tem uporabimo Laplaceovo glajenje:

$$P(w_k|w_{k-N+1}^{k-1}) = \frac{count(w_{k-N+1}^{k-1}w_k) + 1}{count(w_{k-N+1}^{k-1}) + V} \tag{4.10}$$

Vrednost V pri tem predstavlja velikost besednjaka. Model neumnega sestopanja deluje tako, da najprej uporabi 3-gramski jezikovni model, če je frekvenca danega 3-grama večja od 0, drugače uporabi 2-gramski model in tako naprej do 1-gramskega modela. Pri vsakem sestopu izračunano vrednost kaznujemo s faktorjem α. Enačba za izračun vrednosti je sledeča:

$$S(w_k|w_{k-N+1}^{k-1}) = \begin{cases} \frac{count(w_{k-N+1}^{k})}{count(w_{k-N+1}^{k-1})} & ce\ je\ count(w_{k-N+1}^{k}) > 0 \\ \alpha S(w_k|w_{k-N+2}^{k-1}) & drugace \end{cases} \quad (4.11)$$

Pri tem smo vrednost α določili pri 0.4, ki je bila predlagana v literaturi [7].

Za testiranje različnih modelov pri popravljanju črkovanja smo pripravili učno množico, ki je vsebovala 13,000 pravilnih[5] povedi iz našega korpusa komentarjev. Prav tako smo pripravili testno množico, ki je vsebovala 98 povedi, ki so imele eno izmed besed narobe zapisano. Matrike pogostosti različnih napak smo pridobili iz Norvigove tabele pogostosti napak[6]. Rezultati črkovalnih sistemov so prikazani v tabeli 4.9.

Jezikovni model	Točnost
Laplaceov 1-gramski	51%
Laplaceov 2-gramski	63%
Neumno sestopanje	65%

Tabela 4.9: Rezultati popravljanja črkovanja pri uporabi različnih jezikovnih modelov.

4.7 Oblikoslovno označevanje besedila

Oblikoslovno označevanje besedila (angl. *part-of-speech tagging, POS tagging*) je metoda, kjer besedam v besedilu dodelimo skladenjske oznake (samostalnik, pridevnik, glagol, itd.). Pri analizi sentimenta nam dajo oblikoslovne oznake dodatno informacijo v primerih, ko ima lahko ena beseda v različnih kontekstih nasprotno mnjenjsko polarnost. V teh primerih ima beseda tudi velikokrat različno skladenjsko oznako. Tabela 4.10 prikazuje nekaj takšnih primerov iz leksikona Harvard General Inquirer [4].

[5]Pri izboru povedi smo uporabili uveljavljen sistem za pregledovanje črkovanja v besedilih.

[6]http://norvig.com/ngrams/count_1edit.txt

Beseda	Skladenjska oznaka 1	Polarnost	Skladenjska oznaka 2	Polarnost
fine	pridevnik	pozitivna	samostalnik	negativna
fun	samostalnik	pozitivna	glagol	negativna
hit	glagol	pozitivna	pridevnik	negativna
matter	glagol	pozitivna	pridevnik	negativna

Tabela 4.10: Primeri istih besed z različnimi skladenjskimi oznakami in polarnostmi.

Pri oblikoslovnem označevanju smo uporabili metodo spodbujevalnega učenja s povprečnim perceptronom (angl. *averaged perceptron*) [11] in je trenutno tudi privzeta metoda v popularnih orodjih za obdelavo besedil[7]. Pri njej vsak stavek pretvorimo v niz besed in nato iz vsake besede w_i tvorimo atribute, ki so prikazani v tabeli 4.11. Ti atributi so potem uporabljeni pri učenju in napovedovanju skladenjskih oznak.

Ker v našem korpusu komentarjev nismo imeli oblikoslovnih oznak, smo uporabili del korpusa Penn Treebank[8] in dosegli 96% točnost na testni množici. Ker nas je zanimalo, kako dobro se obnese tudi na primerih izven učne domene, smo model nato testirali tudi na delu Brownovega korpusa[9], kjer je pravilno označil 91% besed.

4.8 Klasifikacija z nadzorovanim učenjem

Klasifikacija besedila z nadzorovanim učenjem je postopek, kjer želimo vhodna besedila klasificirati na podlagi besed, ki so prisotne v besedilu. Vhodno besedilo je zelo pogosto predstavljeno z vrečo besed, ki vsebuje besede, prisotne v vhodnem besedilu, in njihovo frekvenco. Klasifikacija teksta z nadzorovanim učenjem je v literaturi obežno opisana in definirana v [20] in jo bomo opisali tudi tu.

[7] http://www.nltk.org/

[8] https://www.cis.upenn.edu/ treebank/

[9] http://www.nltk.org/nltk_data/

Atribut	Primer
w_i	good
zadnje tri črke w_i	ood
prva črka w_i	g
skladenjska oznaka w_{i-1}	glagol
skladenjska oznaka w_{i-2}	samostalnik
skladenjski oznaki w_{i-1} in w_{i-2}	"glagol samostalnik"
skladenjska oznaka w_{i-1} in beseda w_i	"glagol good"
w_{i-1}	is
zadnje tri črke w_{i-1}	is
w_{i-2}	today
w_{i+1}	to
zadnje tri črke w_{i+1}	to
w_{i+2}	run

Tabela 4.11: Seznam atributov besede w_i pri oblikoslovnem označevanju. Pri tem smo uporabili stavek "today is a good day to run", kjer je obravnavana beseda $w_i = good$.

Pri dani množici vhodnih besedil $\mathbb{D}$ in množici razredov $\mathbb{C}$ želimo določiti klasifikacijsko funkcijo γ, ki vhodna besedila klasificira v pripadajoče razrede:

$$\gamma : \mathbb{D} \to \mathbb{C} \tag{4.12}$$

$\mathbb{D}$ je ponavadi predstavljen kot višjedimenzijski prostor dokumentov in $\mathbb{C}$ je množica razredov, ki je vnaprej definirana glede na problem, ki ga rešujemo. Pri tem nam je dana učna množica primerov $\mathbb{L}$, ki je sestavljena iz parov $< d, c >$, kjer je $d \in \mathbb{D}$ in $c \in \mathbb{C}$. Primer para iz učne množice je $< d, c >=<$ *this is awesome app, exuberant* $>$, kjer je prvi element niz besed, drugi element pa ime razreda, ki mu dokument pripada. Klasifikacijsko funkcijo γ izračunamo s pomočjo nadzorovane učne metode η, ki prejme učno množico $\mathbb{L}$ in vrne klasifikacijsko funkcijo γ: $\eta : \mathbb{L} \to \gamma$. Nadzorovana metoda ji pravimo zato, ker potrebuje predhodno klasificirane dokumente v pripadajoče razrede, na katerih se potem uči. Cilj klasifikacije je pridobiti klasifikator, ki ima

visoko točnost tako na testni množici kot na čisto novih podatkih. Pri gradnji našega sistema smo se odločili, da bomo implementirali in testirali tri različne klasifikatorje: naivni Bayesov klasifikator, metodo podpornih vektorjev (angl. *support vector machine, SVM*) in logistično regresijo.

4.8.1 Binarni multinomski naivni Bayes

Družina klasifikatorjev, ki temelji na uporabi Bayesovega teorema, spada med verjetnostne klasifikatorje [20]. Pri analizi sentimenta se najpogosteje najbolje odreže binarni multinomski model [3], ki ga ne smemo zamešati z Bernoullijevim modelom. Celotna družina je poznana kot zelo robustna in njene metode so zelo pogosto uporabljene pri nadzorovanem učenju. Klasifikatorji se obnesejo dobro na velikem razponu problemov, zato so velikokrat uporabljeni kot izhodišče za primerjavo z bolj kompleksnimi metodami. Pri tej metodi določamo verjetnost, da besedilni dokument d spada v razred c in je izračunana kot:

$$P(c|d) \propto P(c) \prod_{1 \leq k \leq n_k} P(t_k|c) \tag{4.13}$$

Pri tem je $P(t_k|c)$ pogojna verjetnost, da se atribut t_k iz d pojavi v dokumentu razreda c in $P(c)$ je apriorna verjetnost razreda c. Če pogojne verjetnosti atributov ne prispevajo dovoljne informacije o tem, v kateri razred naj bi spadalo vhodno besedilo, potem se odločimo za tistega, ki ima višjo apriorno verjetnost. Naš cilj je določiti razred c_{map} (angl. *maximum a posteriori*), ki je najbolj verjeten za besedilni dokument:

$$c_{map} = \underset{c \in \mathbb{C}}{\operatorname{argmax}} P(c|d) \tag{4.14}$$

$$= \underset{c \in \mathbb{C}}{\operatorname{argmax}} \frac{P(d|c)P(c)}{P(d)} \tag{4.15}$$

$$= \underset{c \in \mathbb{C}}{\operatorname{argmax}} P(d|c)P(c) \tag{4.16}$$

$$= \underset{c \in \mathbb{C}}{\operatorname{argmax}} P(c) \prod_{1 \leq k \leq n_k} P(t_k|c) \tag{4.17}$$

Pri tem v koraku (4.16) odstranimo imenovalec, ker je pri izračunu verjetnosti za vse razrede enak. Prav tako pri uporabi te metode naredimo dve predpostavki:

Predpostavka o pogojni neodvisnosti: vsi atributi $d =< t_1, t_2, ..., t_n >$ v d so pogojno neodvisni pri danem razredu c:
$P(t_1, t_2, t_3, ...t_n) = P(t_1|c)P(t_2|c)P(t_3|c)...P(t_n|c)$.

Predpostavka o lokacijski neodvisnosti: pri izračunu $P(d|c)$ predpostavljamo model vreče besed, kjer je informacija o lokaciji atributa znotraj dokumenta izgubljena, in predpostavljamo, da ni pomembna.

Apriorne verjetnosti razredov $P(c)$ izračunamo s pomočjo ocene maksimalnega verjetja (angl. *maximum likelihood estimation*):

$$P(c) = \frac{N_c}{N}, \tag{4.18}$$

kjer N_c predstavlja število dokumentov, ki pripadajo razredu c v učni množici, N pa število vseh dokumentov v učni množici. Za izračun $P(d|c)$ moramo pri računanju ocene maksimalnega verjetja dodati še Laplaceovo glajenje (angl. *Laplace smoothing*):

$$P(t|c) = \frac{\#t_c + 1}{\sum\limits_{t'_c \in V} (\#t' + 1)} = \frac{\#t_c + 1}{(\sum\limits_{t'_c \in V} \#t') + B}, \tag{4.19}$$

kjer je V besednjak in B število vseh atributov v V: $B = |V|$. $\#t_c$ je število pojavitev atributa t v učni množici razreda c in $\sum_{t'_c \in V}(\#t')$ je število pojavitev vseh atributov iz V, ki pripadajo razredu c. Laplaceovo glajenje dodamo tako, da atributu v števcu in vsem atributom v vsoti vrste v imenovalcu prištejemo ena. Razlog za glajene izhaja iz problema, ko moramo izračunati pogojno verjetnost nekega atributa, ki ga ni v učni množici. Brez glajenja bi bila verjetnost $P(t|c)$ v tem primeru 0 in posledično tudi verjetnost razreda $P(c|d)$ 0, in to ne glede na verjetnosti ostalih atributov v dokumentu d.

Pri implementaciji našega sistema smo se odločili za binarni multinomski model, ker je na predhodnih testih pri analizi sentimenta dal najboljše rezultate. Pri tem smo testirali navaden, multinomski ter Bernoullijev model. Pri binarnem modelu so vrednosti atributov binarne, tako kot pri Bernoullijevem modelu in predstavljajo prisotnost besede v dokumentu. Pri navadnem multinomskem modelu pa so vrednosti atributov diskretne in ponavadi predstavljajo število pojavitev besed v dokumentu. Podrobnosti Bernoullijevega modela tu ne obravnavamo, je pa obsežno opisan v literaturi [20].

4.8.2 Logistična regresija

Model logistične regresije je verjetnostni in linearni klasifikator in za razliko od naivnega Bayesa spada med diskriminantne modele [14]. To pomeni, da pri računanju $P(y|x)$ najprej diskriminira med različnimi vrednostmi razreda y, namesto da bi prvo izračunal $\hat{y} = \text{argmax}_y P(y|x)$. Model logistične regresije oceni vrednost $P(y|x)$ tako, da vzame nekatere vhodne atribute in jih linearno kombinira: vsakega pomnoži z neko pripadajočo utežjo in jih sešteje. Nad to kombinacijo nato aplicira funkcijo. Tako je enačba (4.20) za računanje verjetnosti, da je y vrednost razreda c, pri podanem vhodnem vektorju x enaka:

$$P(c|x) = \frac{exp\Big(\sum_{i}^{N} w_i f_i(c,x)\Big)}{\sum_{c' \in C} exp\Big(\sum_{i}^{N} w_i f_i(c,x)\Big)} \tag{4.20}$$

Pri tem v izračunu uporabimo eksponentno funkcijo, ki poskrbi, da so vrednosti produkta uteži in atributov pozitivne ter normalizacijo (imenovalec), da dobimo veljavne verjetnostne vrednosti med 0 in 1. Atributi f_i prav tako niso samo lastnosti x, ampak so njihove vrednosti, odvisne od x in c.

Na primer, da delamo klasifikacijo besedila in da nas zanima, ali vhodno besedilo izraža pozitivno ali negativno mnenje. Pri tem imamo dane tri potencialne atribute:

$$f_1(c,x) = \begin{cases} 1 & \text{"}awesome\text{"} \in x \wedge c\ je\ pozitiven \\ 0 & sicer \end{cases}$$

$$f_2(c,x) = \begin{cases} 1 & \text{"}bad\text{"} \in x \wedge c\ je\ negativen \\ 0 & sicer \end{cases}$$

$$f_3(c,x) = \begin{cases} 1 & \text{"}great\text{"} \in x \wedge c\ je\ negativen \\ 0 & sicer \end{cases}$$

Vsak od teh atributov ima pripadajočo utež. Tako lahko sklepamo, da bi morali biti uteži za f_1 in f_2 pozitivni in utež za f_3 negativna (ker ima beseda "great" ponavadi pozitivno orientacijo). Uteži predstavljajo pomembnost posameznega atributa za nek razred.

Pri klasifikaciji tako samo izračunamo verjetnost za vsak razred in izberemo tistega, ki ima največjo. Pri tem lahko imenovalec v (4.20) in eksponentno funkcijo ignoriramo in iskanje maksimalne vrednosti poenostavimo v enačbi (4.21).

$$\begin{aligned} \hat{c} &= \underset{c \in C}{\operatorname{argmax}}\, P(c|x) \\ &= \underset{c \in C}{\operatorname{argmax}}\, \frac{exp\Big(\sum_i^N w_i f_i(c,x)\Big)}{\sum_{c' \in C} exp\Big(\sum_i^N w_i f_i(c,x)\Big)} \\ &= \sum_i^N w_i f_i(c,x) \end{aligned} \tag{4.21}$$

Izračun uteži določimo s pogojno oceno največjega verjetja (angl. *conditional maximum likelihood estimation*). To pomeni, da izberemo parametre w tako, da maksimizirajo verjetnosti razredov y v učnih podatkih. Na podlagi celotne učne množice bi tako optimalne uteži bile:

$$\hat{w} = \underset{w}{\operatorname{argmax}} \sum_j logP(y_j|x_j) \tag{4.22}$$

Pri tem določimo funkcijo $L(w)$, katere vrednost želimo maksimizirati:

$$L(w) = \sum_j logP(c_j|x_j) \tag{4.23}$$

$$= log \sum_j exp\Big(\sum_i^N w_i f_i(c,x)\Big) - \tag{4.24}$$

$$log \sum_j \sum_{c' \in C} exp\Big(\sum_i^N w_i f_i(c,x)\Big) \tag{4.25}$$

$$\tag{4.26}$$

Iskanje primernih uteži pri tem postane konveksno optimizacijski problem in za njegovo rešitev uporabimo eno od metod vzpenjanja (angl. *hill climbing metod*). Ena izmed teh metod je stohastično gradientno vzpenjanje (angl. *stohastic gradient ascending*), ki začne z ničelnimi utežmi in se potem premika v smeri odvoda $L'(w)$.

$$L'(w) = \sum_i f_k(c_i, x_i) - \sum_i \sum_{c' \in C} P(c'|x_i) f_k(c'_i, x_i) \tag{4.27}$$

$$= observed\ count(f_k) - expected\ count(f_k) \tag{4.28}$$

$$\tag{4.29}$$

Pri računanju tega odvoda (4.27) je *expected count* dejansko število pojavitev atributa f_k v učnih podatkih in *observed count* število pojavitev atributa s strani trenutnega modela in njegovih verjetnosti. Pri optimalnih utežeh modela so dobljene vrednosti ponovitev atributov enake tistim, ki jih imamo v učunih podatkih. Pri teh utežeh, ki model idealno prilagodijo učnim podatkom, nastane problem. Tako je atribut, ki se pojavi v samo enem razredu, idealen za napovedovanje tega razreda in bo tako dobil veliko utež. Proces določanja uteži bo stremel k čim boljšemu prileganju uteži k učnim podatkom in s tem lahko dobimo model, ki se učnim podatkom preveč prilega

(*overfitting*). Da se temu problemu izognemo, dodamo v enačbo za izračun uteži (4.22) regularizacijsko funkcijo:

$$\hat{w} = \underset{w}{\operatorname{argmax}} \sum_j logP(y_j|x_j) - \alpha R(w) \tag{4.30}$$

Pri tem $\alpha R(w)$ v 4.30 predstavlja kazensko vrednost za uteži, ki so prevelike. Obstajata dva načina kaznovanja uteži:

L2 regularizacija : predstavlja kvadratno funkcijo nad utežmi in je enaka evklidski razdalji: $R(w) = \sum_i^N w_i^2$

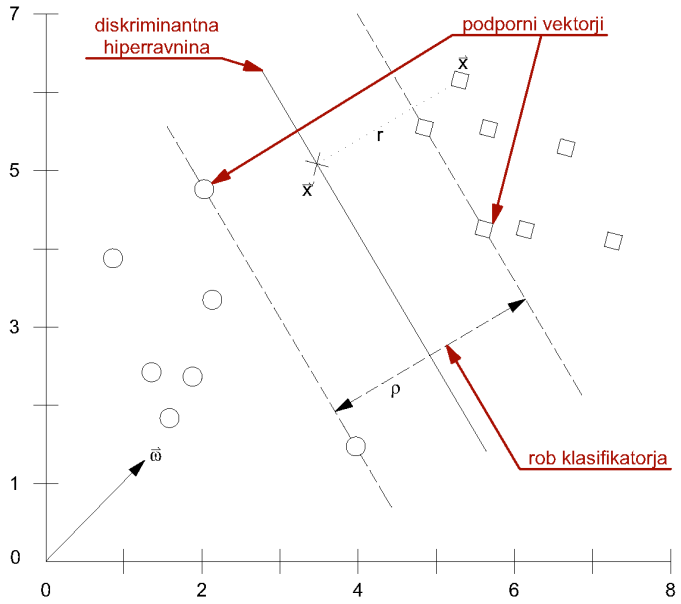
L1 regularizacija : je poznana kot prva norma (angl. *Manhattan distance*): $R(w) = \sum_i^N w_i$

Regularizacijski parameter α pa predstavlja kompromis med tem, kako dobro želimo, da se uteži prilegajo učnim podatkom, in našo željo, da uteži niso prevelike. Če nastavimo parameter na preveliko vrednost, bo model preveč splošen in se bo učnim podatkom premalo prilegal (angl. *underfitting*).

4.8.3 Metoda podpornih vektorjev

Metoda podpornih vektorjev optimizira iskanje diskriminantne hiperravnine, ki loči primere dveh razredov [20]. Če sta dva razreda linearno ločljiva, potem med njima obstaja nešteto mnogo možnih razmejitev. Zato je cilj te metode najti takšno razmejitev, ki je maksimalno oddaljena od primerov iz obeh razredov. Ta maksimalna razdalja je imenovana rob klasifikatorja. S tem je odločitvena funkcija SVM določena s podmnožico učnih vektorjev, ki definirajo pozicijo tega roba. Tem vektorjem pravimo podporni vektorji. Maksimalen rob omogoči klasifikatorju, da majhna napaka ali variacija v vhodnih dokumentih še ne povzroči klasifikacije v napačen razred.
Diskriminantno hiperravnino definiramo kot:

$$\vec{w}\vec{x} = -b \tag{4.31}$$



Slika 4.1: Diskriminantna hiperravnina in podporni vektorji.

Kjer je $\vec{w}$ vektor uteži diskriminantne hiperravnine, b predstavlja odmik in $\vec{x}$ je vektor vrednosti atributov, ki predstavljajo točko na hiperravnini. Pri tej metodi sta klasifikacijska razreda vedno opredeljena z vrednostima -1 in 1. Tako lahko definiramo linearno klasifikacijsko funkcijo kot:

$$f(\vec{x}) = sign(\vec{w}\vec{x}) + b \tag{4.32}$$

Pri tem so vsi primeri, katerih rešitev zgornje enačbe je negativna, klasificirani v razred -1 in v 1 za pozitivne primere. Zaupanje v klasifikacijsko odločitev raste z razdaljo točke od naše hiperravnine. Bolj kot je točka oddaljena od nje, večje imamo zaupanje v pravilnost napovednega razreda. Zato želimo določiti takšno hiperravnino, kjer so točke kar najbolj oddaljene od nje. Razdaljo posameznega vektorja $\vec{x}$ izračunamo kot:

$$r = y\frac{\vec{w}\vec{x} + b}{|\vec{w}|} \tag{4.33}$$

y predstavlja predznak oziroma vrednost razreda podanega vektorja. Vektorji, ki so najbližje diskriminantni hiperravnini, so podporni vektorji in razmejitveni pas med podpornimi vektorji obeh razredov je določen kot dvakratnik razdalje r teh vektorjev od hiperravnine. Naš cilj je maksimizirati razmejitveni pas ρ, ki postane kvadratičen optimizacijski problem in rešitev vključuje konstrukcijo dualnega problema in Lagrangeovih multiplikatorjev. Za probleme v klasifikaciji teksta, ki je visoko dimenzionalen, ponavadi podatki niso linearno ločljivi. V tem primeru lahko dopustimo, da nekaj učnih podatkov pade v razmejitveni pas in klasifikator naredi nekaj napak. Pri tem je klasifikator kaznovan in kazen je odvisna od razdalje med napačno klasificiranim podatkom in robom klasifikatorja. Optimizacijski problem postane iskanje kompromisa med širino razmejitvenega pasu in številom napak na učni množici. Pri tem vpeljemo regularizacijski parameter in z njim lahko določamo stopnjo prevelike in premajhne naučenosti.

4.9 Vrednotenje klasifikatorjev

Ko imamo klasifikatorje zgrajene, nas zanima, kako točni so rezultati pri klasifikaciji komentarjev. Odločili smo se za uporabo klasifikacijske točnosti (angl. *classification accuracy, CA*), preciznosti (angl. *precision*), priklica (angl. *recall*) in mere F1 (angl. *F1 measure*). Pri vrednotenju smo uporabili metodo notranjega prečnega preverjanja (angl. *internal cross-validation*). Postopek je sledeč:

1. Označene primere razdelimo na k množic. Pri tem $k-1$ množic uporabimo za učenje in eno za testiranje našega klasifikatorja. Pri tem množico za testiranje označimo z A in množico za učenje z B. Postopek ponovimo k-krat, kjer v vsaki iteraciji uporabimo drugo množico za testiranje.

2. V vsaki iteraciji iz prejšnje točke, množico B razdelimo na k množic. Pri tem $k-1$ množic uporabimo za učenje in eno za testiranje mere F1

pri različnih izborih atributov in vrednostih regularizacijskih parametrov. Postopek ponovimo k-krat. Na koncu izberemo najbolj primeren regularizacijski parameter in atribute, ki so bili v povprečju najbolje ocenjeni.

3. Na koncu uporabimo celotno množico B in naučimo klasifikator pri izbranih parametrih in naboru iz prejšnje točke. Končni rezultat iteracije dobimo s testiranjem klasifikatorja na množici A.

Na koncu izvedemo tudi Wilcoxonov test predznačenih rangov (angl. *Wilcoxon signed ranks test*), da vidimo, ali obstajajo statistične razlike med uspešnostmi klasifikatorjev.

4.9.1 Klasifikacijska točnost

Klasifikacijska točnost pove [1], kako dobro je klasifikator označil vhodna besedila. Izračunamo jo kot razmerje med vsoto pravilno klasificiranih primerov in vsemi primeri.

$$Accuracy = \frac{T_P + T_N}{T_P + T_N + F_P + F_N} \tag{4.34}$$

1. T_P: število pravilno klasificiranih primerov prvega (pozitivnega) razreda.

2. T_N: število pravilno klasificiranih primerov drugega (negativnega) razreda.

3. F_P: število nepravilno klasificiranih primerov prvega (pozitivnega) razreda.

4. F_N: število nepravilno klasificiranih primerov drugega (negativnega) razreda.

V enačbi (4.34) T_P in T_N, predstavljata število pravilno klasificiranih primerov iz obeh razredov in $T_P + T_N + F_P + F_N$ število vseh klasificiranih

primerov. Problem pri uporabi točnosti je, da se v določenih primerih ne izkaže kot dober kriterij. Na primer, da imamo učno množico, kjer je porazdelitev primerov 8:2 v prid enemu izmed razredov. V tem primeru bo kriterij kazal 80% točnost že za preprost klasifikator, ki vsak primer klasificira v večinski razred. Zaradi tega potrebujemo druge mere, ki nam bodo dali več vpogleda v končno razdelitev v razrede.

4.9.2 Preciznost, priklic in mera F

Preciznost je kriterij [14], ki nam pove, kolikšen delež primerov napovednega razreda je pravilno klasificiranega. Izračunamo jo kot:

$$Precision = \frac{T_P}{T_P + F_P}, \tag{4.35}$$

kjer so definicije T_P in F_P podane v enačbi (4.34). Priklic pa nam pove delež pravilno klasificiranih primerov dejanskega razreda. Izračunamo ga kot:

$$Recall = \frac{T_P}{T_P + F_N} \tag{4.36}$$

Priklic in preciznost sta uporabna kriterija, ki jih ponavadi združimo v kombiniran kriterij, ki ga imenujemo F mera (angl. *F measure*). F mera predstavlja uteženo harmonično sredino [20] in je zelo konzervativna povprečna ocena. Izračun je prikazan v (4.37).

$$F = \frac{1}{\alpha \frac{1}{Precision} + (1-\alpha)\frac{1}{Recall}} = \frac{(\beta^2+1)Precision \cdot Recall}{\beta^2 Precision + Recall} \tag{4.37}$$

Pri ocenjevanju bomo uporabili uravnoteženo verzijo, ki je poznana kot mera F1. Pri meri F1 nastavimo parameter β na 1, oziroma parameter α na $\frac{1}{2}$. Tako dobimo končen zapis formule za izračun mere F1 v (4.38).

$$F1 = \frac{2Precision \cdot Recall}{Precision + Recall} \tag{4.38}$$

4.9.3 Wilcoxonov test predznačenih rangov

Wilcoxonov test predznačenih rangov spada med neparametrične statistične [6, 10] teste in ga bomo uporabili za primerjavo klasifikatorjev. Pri tem bomo testirali ničelno statistično hipotezo H_0, ki pravi, da med rezultati klasifikatorjev ni statistično značilnih razlik na stopnji značilnosti p. Prav tako bomo postavili alternativno hipotezo H_1, ki pravi, da statistično značilne razlike so. Če bodo vzorčni podatki preveč odstopali od H_0, bomo rekli, da so razlike statistično značilne in bomo hipotezo H_0 zavrnili. Wilcoxonov test bomo opravili z naslednjimi koraki:

1. Pripravili bomo množico meritev v velikosti N, v kateri bodo pari posameznih meritev dveh klasifikatorjev na istih učnih in testnih podatkih. Pri tem prvo meritev iz para i označimo z $x_{1,i}$ in drugo z $x_{2,i}$.

2. Izračunamo $y_i = x_{1,i} - x_{2,i}$ za $i = 1, ..., N$.

3. Če smo pri katerem izmed izračunov dobili vrednost 0, potem ta vzorec ignoriramo. S tem se naša množica meritev zmanjša na M.

4. Rezultate y_i rangiramo po njihovi absolutni vrednosti od 1 do M. Pri tem rezultat z najnižjo absolutno vrednostjo dobi rang 1 in tisti z najvišjo rang M. Če imamo več rezultatov, ki imajo enako absolutno vrednost, potem ti rezultati dobijo rang, ki je enak povprečju njihovih rangov. Na primer, da imamo dva najnižja rezultata 1 in -1. Pri tem bi eden dobil rang 1 in drugi rang 2, vendar, ker imata enako absolutno vrednost, obema pripada povprečen rang 1.5.

5. Ločeno seštejemo vrednosti rangov pozitivnih rezultatov in negativnih rezultatov. Izberemo tisti seštevek, ki ima manjšo vrednost, in ga označimo z W.

6. Uporabimo tabelo 4.12, ki vsebuje Wilcoxonove kritične vrednosti. Stolpec v tabeli N predstavlja število vzorcev M in posamezna vrstica predstavlja vrednost praga pri različnih stopnjah značilnosti. Če je W

manjši ali enak od vrednosti praga p, potem rečemo, da so razlike med rezultati klasifikatorjev statistično pomembne pri stopnji značilnosti p. In v tem primeru bomo hipotezo H_0 zavrnili.

N	$p = 0.05$	$p = 0.02$	$p = 0.01$
6	0	-	-
7	2	0	-
8	4	2	0
9	6	3	2
10	8	5	3
...	...	...	...
24	81	69	61
25	90	77	68
26	98	85	76
27	107	93	84
28	117	102	92
29	127	111	100
30	137	120	109

Tabela 4.12: Wilcoxonova tabela kritičnih vrednosti. Stolpec N predstavlja število vzorcev, ostali stolpci pa prag za izračunano vrednost W. Če je W manjša ali enaka od vrednosti praga, potem je razlika med rezultati klasifikatorjev statistično pomembna pri stopnji značilnosti p.

Statistične teste bomo opravili na vrednostih mere F1 pri notranjem prečnem preverjanju različnih klasifikatorjev na istih vhodnih množicah.

4.10 Predstavitev in izbor atributov

V prejšnjih poglavjih smo si ogledali postopek pretvorbe vhodnega besedila v vrečo besed, ki ima žetone in njihove vrednosti v primerni obliki za nadaljnjo vektorizacijo. Zgradili smo 8 različnih naborov atributov. Osnoven potek pretvorbe vhodnega besedila za vse nabore povzamemo v naslednjih korakih:

Razčlenjevanje: proces razčlenjevanja vhodnega niza z uporabo vseh metod opisanih v razdelku 4.1;

Normalizacija in negacija: primerno normaliziramo in negiramo vhodni niz z uporabo korakov v razdelku 4.2;

Generiranje n-gramov: pri generiranju n-gramov smo se glede na predhodne empirične teste odločili za 1-grame in 3-grame na nivoju vhodnega besedila ter 3-grame na nivoju besede;

Frekvenca prisotnosti žetonov: pri izbiri metode za naš sistem smo se odločili za uporabo binarne prisotnosti na podlagi predhodnih testov.

Ocena komentarja: ocena komentarja je vnaprej določena s strani avtorja in jo pridobimo iz vhodnega dokumenta. Kot smo prikazali v poglavju 3, so v veliki večini navdušujoči komentarji ocenjeni z oceno 5. Pri tem vrednost ocene binariziramo na podlagi ročnega pravila, ki oceno 5 preslika v vrednost 1 in vse ostale v vrednost 0. S tem tudi dosežemo približno enakomerno porazdelitev dokumentov za obe vrednosti atributov.

Besedila, zapisana z velikimi črkami: ker v procesu razčlenjevanja normaliziramo vse besede v besede, zapisane z malimi črkami, s tem izgubimo del informacije o intenziteti mnenja. Zato to informacijo dodamo v naš nabor atributov naknadno, kjer označimo komentar, ki vsebuje besede, zapisane z velikimi črkami, z 1 in preostale z 0.

Rezultat pretvorbe besedila, ki ga uporabimo pri vseh naborih, je prikazan v tabeli 4.13.

V prejšnjih poglavjih opisali tudi nekaj bolj naprednih metod obdelave besedil in analize sentimenta, ki smo jih uporabili pri gradnji različnih naborov atributov. Pri tem smo uporabili naslednje metode:

Korenjenje: vse besede v vhodnem komentarju korenimo s pomočjo Porterjevega algoritma, kot je opisano v razdelku 4.5.

Skupina atributov	Primeri	Vrednost
besedni 1-grami	“awesome”, “bad”, “not_NEG”	binarna
besedni 3-grami	“really great app”	binarna
znakovni 3-grami	“awe”, “wes”, “eso”, “som”, “ome”	binarna
ocena komentarja	review_rating	binarna
komentarji z velikimi črkami	review_case	binarna

Tabela 4.13: Nabor osnovnih atributov. Za oceno komentarja in komentarje z velikimi črkami tvorimo posebne žetone “review_rating” in “review_case”.

Popravljanje črkovanja: Pri povedih v vhodnih komentarjih preverimo črkovanje. Tiste povedi, ki vsebujejo napake, skušamo popraviti s pomočjo neumnega sestopanja, ki smo ga opisali v razdelku 4.6. Algoritem predvideva, da je v posamezni besedi prisotna ena napaka in poljubno število napak v sami povedi.

Označevanje besedila: Vse besede v komentarju oblikoslovno označimo s pomočjo metode, opisane v poglavju 4.7. Pri tem besedne oznake pripnemo k samim besedam.

Označevanje lokacije besede: Pri tej metodi označimo posamezno besedo glede na njeno lokacijo v besedilu. Podatek o lokaciji podamo v eni izmed treh vrednosti: začetek, sredina ali konec komentarja. Podatek pripnemo k sami besedi.

Uporaba mnenjskega leksikona: število negativnih in pozitivnih besed v vhodnem komentarju bomo izračunali s pomočjo leksikona[10]. Nato bomo dodali atribut, ki bo postavljen na 1, če komentar ne bo vseboval negativnih besed in bo imel vsaj eno pozitivno besedo. V nasprotnem

[10]Hu and Liu’s lexicon: https://www.cs.uic.edu/ liub/FBS/sentiment-analysis.html

primeru bo atribut nastavljen na 0.

Uporaba sistema VADER: vsak vhodni komentar bomo analizirali s pomočjo sistema VADER, ki bo vrnil dve vrednosti: oceno negativnosti in pozitivnosti komentarja. Pridobljeni vrednosti bosta na intervalu med 0 in 1. Mi ju bomo predstavili kot ločena atributa. Vsak atribut bo imel vrednost 1, če bo ocena nad 0.5, sicer bo vrednost postavljena na 0.

Primere atributov, ki smo jih dobili z zgoraj opisanimi metodami, so prikazani v tabeli 4.14.

Metoda	Primeri atributov	Vrednost
Korenjenje	"awesome" → "awesom"	binarna
Popravljanje črkovanja	"This", "is", "awsome", "gam" → "This", "is", "awesome", "game"	binarna
Označevanje besedila	"This", "is", "awesome" → "This:DT", "is:VBZ", "awesome:JJ"	binarna binarna
Označevanje lokacije besede	"This", "is", "awesome" → "This_BEG", "is_MID", "awesome_END"	binarna
Mnenjski leksikon	lexicon_value	binarna
VADER	vader_pos, vader_neg	binarna

Tabela 4.14: Primeri atributov pri uporabi naprednih metod za obdelavo besedil in določanje sentimenta.

Nabore atributov smo generirali tako, da smo pri vsakem naboru uporabili osnoven potek pretvorbe besedila in eno od naprednejših metod. Poleg tega smo dodali tudi nabor atributov, kjer uporabimo vse napredne metode in nabor, kjer ne uporabimo nobene. Nabori tako zgrajenih atributov so prikazani v tabeli 4.15.

Metode za izbiro atributov nam pomagajo izbrati podmnožico najbolj pomembnih atributov v učni množici. Za učenje in napovedovanje klasifikatorja lahko nato uporabimo le to podmnožico atributov. Z manjšim številom

Nabor atributov	Korenjenje	POS oznake	lokacija besede	Popravljanje črkovanja	Mnenjski leksikon	VADER
All	x	x	x	x	x	x
Stem	x					
POS		x				
LOC			x			
Spell				x		
Lex					x	
Vader						x
Base						

Tabela 4.15: Nabori atributov pri uporabi različnih metod. Vsak nabor atributov smo poimenovali glede na metodo, ki jo uporabimo pri gradnji posameznega nabora. Nabor All uporablja vse napredne metode, nabor Base pa nobene od njih.

atributov bo klasifikator manj kompleksen in učenje ter napovedovanje bo hitrejše. Prav tako s tem preprečimo, da bi se klasifikator preveč prilegal učnim podatkom. Za izbiro atributov bomo uporabili metodo χ^2. V statistiki se uporablja za določanje neodvisnosti dveh dogodkov [20]. Pri izbiri atributov za klasifikacijo teksta sta ta dva dogodka pojavitev besede t in pojavitev razreda c v učni množici in njuno vrednost izračunamo s pomočjo enačbe (4.39).

$$\chi^2(t,c) = \sum_{e_t \in \{0,1\}} \sum_{e_c \in \{0,1\}} \frac{(N_{e_t e_c} - E_{e_t e_c})^2}{E_{e_t e_c}} \tag{4.39}$$

Pri tem e_t določa, ali dokument vsebuje besedo t, in e_c določa ali je dokument v razredu c ali ne. $N_{e_t e_c}$ je opažena vrednost pojavitev v učni množici in $E_{e_t e_c}$ je pričakovana vrednost, pri pogoju, da sta t in c neodvisna. $E_{e_t e_c}$ izračunamo kot:

$$E_{e_t e_c} = N \cdot \frac{N_{e_t 1} + N_{e_t 0}}{N} \cdot \frac{N_{1 e_c} + N_{0 e_c}}{N}, \tag{4.40}$$

kjer N predstavlja celotno število dokumentov v učni množici. Enačba (4.39) tako pove, koliko se pričakovana in opažena vrednost razlikujeta. Velika vre-

dnost χ^2 nakazuje, da je predpostavka o neodvisnosti napačna, saj pojavitev besede v besedilu kaže na večjo ali manjšo verjetnost razreda. Zaradi tega je ta beseda pomembna kot atribut.

4.11 Rezultati

Pri notranjem prečnem preverjanju 10x10 smo na učnih množicah najprej poiskali primerno število atributov za vse tri klasifikatorje za vse nabore atributov. Originalno število atributov v vsakem naboru je presegalo 100,000. Cilj je bil to število v vsakem naboru zmanjšati in pri tem ohraniti čim višjo vrednost mere F1. Za vrednotenje atributov smo uporabili metodo χ^2. Povprečni rezultati notranjega prečnega preverjanja na učnih množicah so prikazani na sliki 4.2. Vidimo lahko, da uspešnost klasifikatorjev narašča do meje približno 1000 atributov, nato pa se ustali. Končni nabor atributov za testiranje učne množice smo zgradili tako, da smo nabore iz vseh učnih množic rangirali in izbrali 1000 najboljših.

Ker metodi podpornih vektorjev in logistične regresije podpirata tudi regularizacijo, ki prepreči preveliko prileganje klasifikatorja k učni množici, smo poiskali tudi primerne regularizacijske parametre. Pri obeh metodah smo za kaznovanje uporabili L2 regularizacijo. Povprečni rezultati notranjega prečnega preverjanja na učnih množicah so prikazani na sliki 4.3. Na slikah 4.4a in 4.4b smo prikazali tudi povprečne rezultate mere F1 na učnih in testnih množicah pri uporabi regularizacije. Iz slik lahko vidimo, da je bila primerna vrednost regularizacijskega parametra 0.1 za metodo podpornih vektorjev in 0.25 za logistično regresijo. Takšne vrednosti smo izbrali, ker najbolje maksimizirajo vrednost mere F1 in pri tem zmanjšajo prileganje klasifikatorja učni množici.

Za vse nabore atributov in klasifikatorjev smo uporabili enake učne in testne podmnožice, kar nam je omogočilo izračun Wilcoxonovega testa predznačenih rangov za primerjavo klasifikatorjev. Končni rezultati, ki prikazujejo povprečno točnost, preciznost, priklic in mero F1 posameznega klasifika-

torja preko vseh naborov atributov so prikazani v tabeli 4.16. Rezultati, ki prikazujejo povprečne vrednosti posameznega nabora atributov preko vseh klasifikatorjev, so prikazani v tabeli 4.17.

Klasifikator	Povprečen CA	Povprečen priklic	Povprečna preciznost	Povprečen F1
NB	0.83	0.91	0.78	0.84
LR	0.92	0.90	0.93	0.92
SVM	0.92	0.90	0.93	0.92

Tabela 4.16: Povprečni rezultati klasifikatorjev preko vseh naborov atributov.

Nabor atributov	Povprečen CA	Povprečen priklic	Povprečna preciznost	Povprečen F1
All	0.89	0.91	0.88	0.89
Stem	0.89	0.91	0.88	0.89
POS	0.89	0.91	0.89	0.89
LOC	0.89	0.90	0.89	0.89
Spell	0.89	0.91	0.88	0.89
Lex	0.88	0.91	0.88	0.89
Vader	0.88	0.91	0.87	0.89
Base	0.89	0.91	0.88	0.89

Tabela 4.17: Povprečni rezultati naborov atributov preko vseh klasifikatorjev.

Wilcoxonove teste predznačenih rangov smo izvedli na rezultatih notranjega prečnega preverjanja pri vseh naborih atributov. Pri vsakem testu smo vzeli enak nabor atributov za vse tri klasifikatorje. Pri vseh naborih atributov je bila razlika v meri F1 klasifikatorjev SVM in naivnega Bayesa statistično pomembna ($W = 0$, $p = 0.01$ za vse nabore). Prav tako je bila razlika v meri F1 klasifikatorjev logistične regresije in naivnega Bayesa statistično pomembna ($W = 0$, $p = 0.01$ za vse nabore). Primerjali smo tudi vse nabore atributov pri uporabi vseh treh klasifikatorjev in dobili statistične

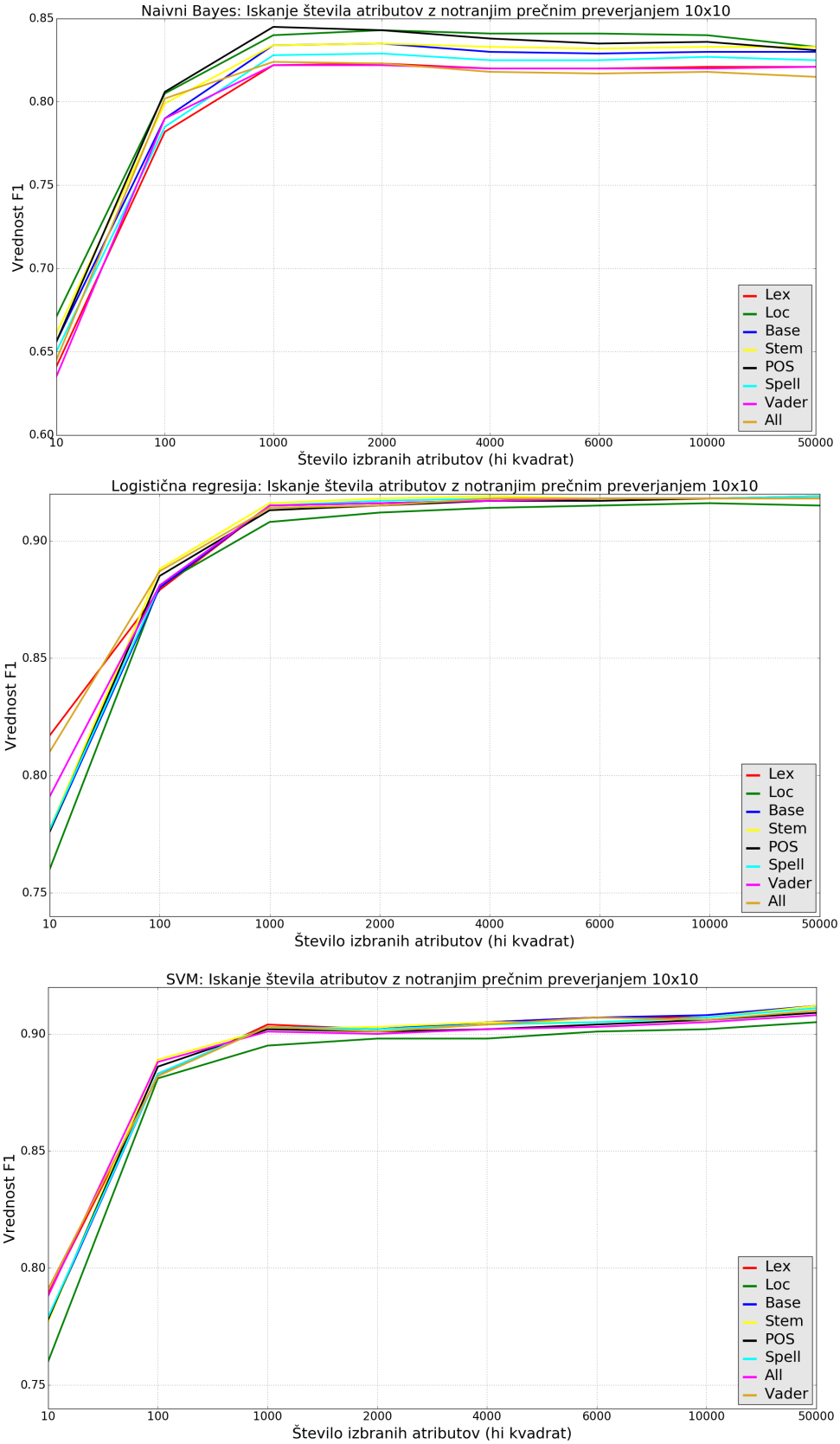
razlike v rezultatih mere F1 pri sledečih:

Nabor POS: nabor se je izkazal za boljšega od naborov LOC ($N = 30$, $W = 128$, $p = 0.05$) in All ($N = 29$, $W = 117$, $p = 0.05$).

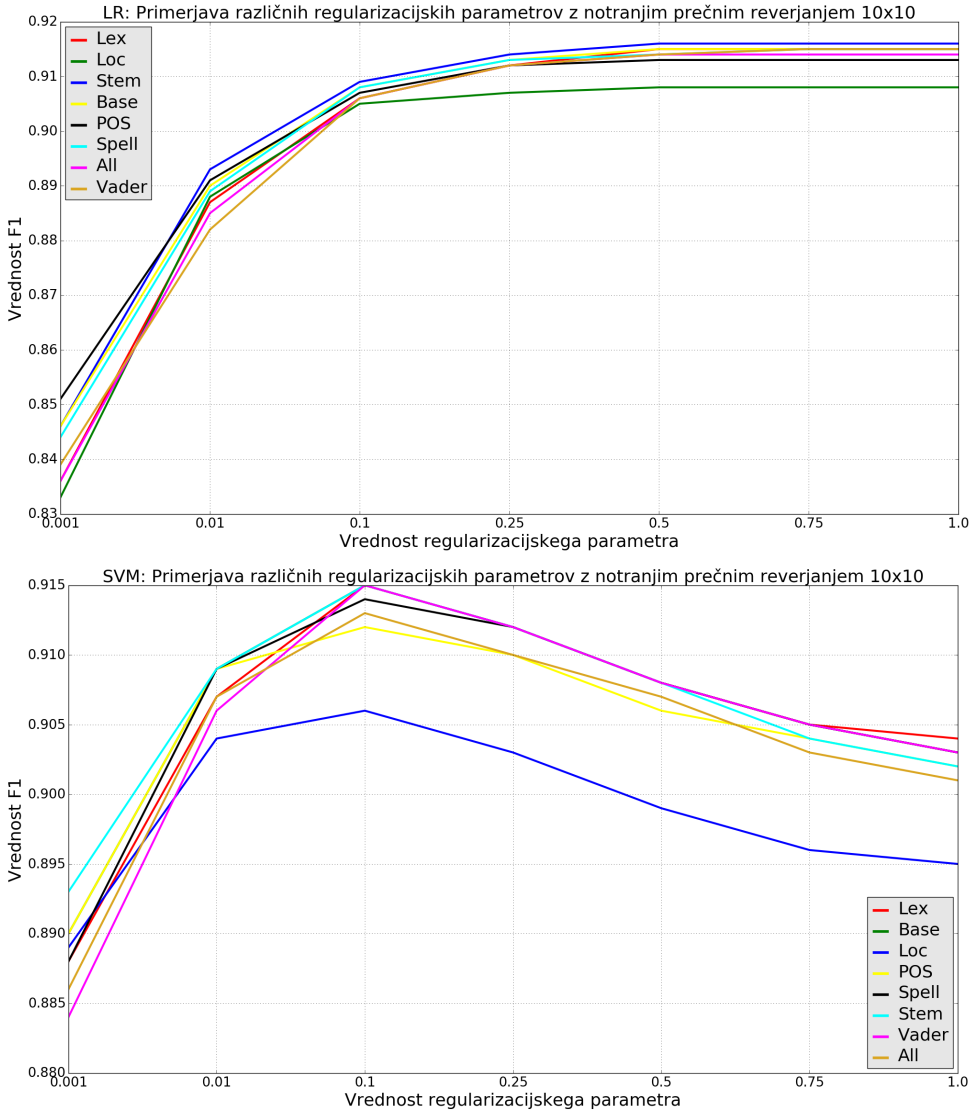
Nabor Stem: nabor je bil boljši od naborov Spell ($N = 29$, $W = 111$, $p = 0.02$) in Vader ($N = 30$, $W = 133$, $p = 0.05$).

Nabor Base: nabor se je izkazal za boljšega od naborov Lex ($N = 24$, $W = 44$, $p = 0.01$), Spell ($N = 30$, $W = 125$, $p = 0.05$) in Vader ($N = 24$, $W = 41$, $p = 0.01$).

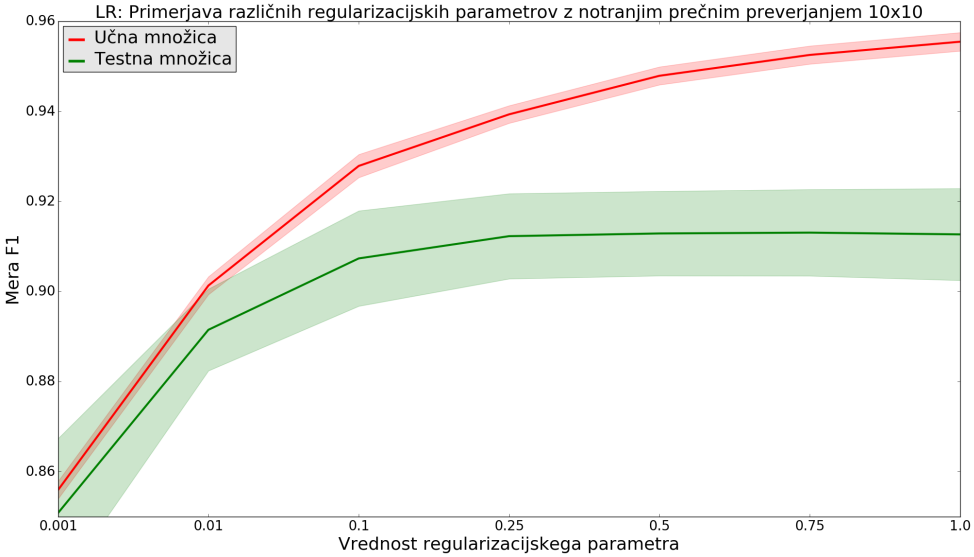
Pri ostalih naborih statističnih razlik ni bilo.



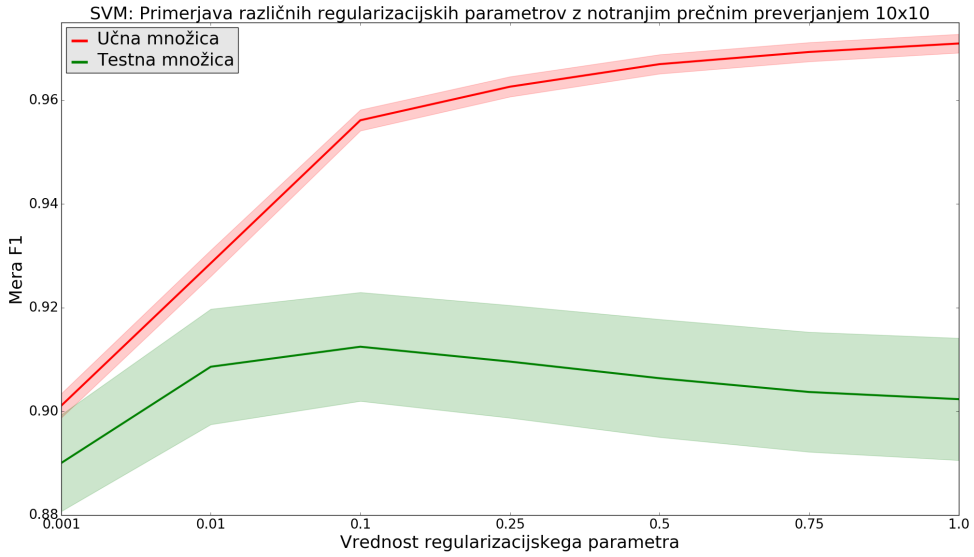
Slika 4.2: Vrednost mere F1 za klasifikatorje pri različnih naborih atributov.



Slika 4.3: Vrednost mere F1 pri različnih regularizacijskih parametrih za logistično regresijo in SVM. Manjša vrednost parametra pomeni večjo regularizacijo.



(a) lahko, da regularizacija vpliva na mero F1, ki pri obeh množicah narašča, nato pa se pri testni množici ustali pri vrednosti 0.25.



(b) Regularizacija vpliva na mero F1 pri obeh množicah in pri testni množici doseže vrh pri 0.1, nato pa začne padati.

Slika 4.4: Vrednost mere F1 na učni in testni množici pri uporabi regularizacije za logistično regresijo in SVM. Pri tem smo uporabili nabor atributov POS. Polna črta predstavlja povprečno vrednost F1, barvni pas okrog črte pa standardni odklon pri rezultatih notranjega prečnega preverjanja.

Poglavje 5

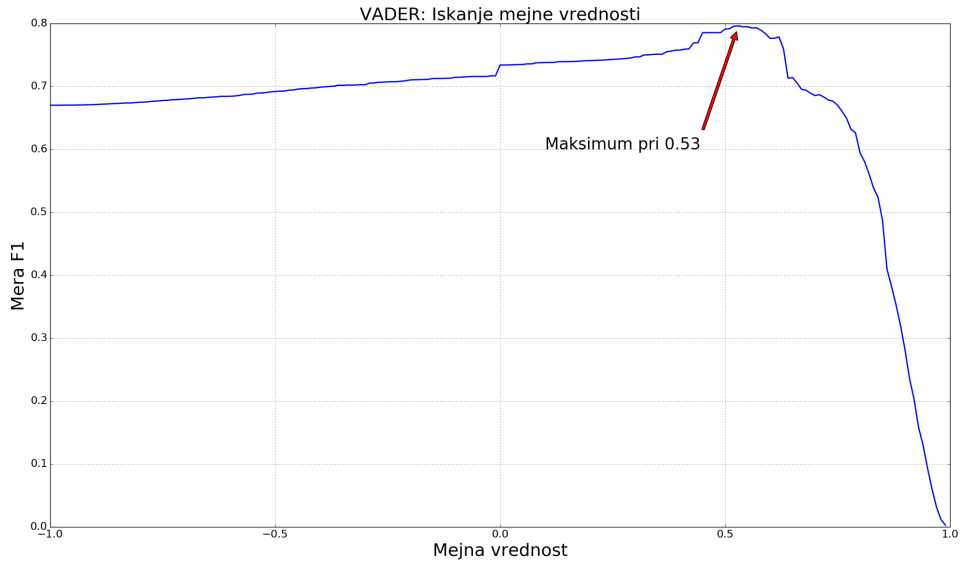
Primerjava z drugimi sistemi

Na koncu primerjamo, kako dobro se obnesejo drugi sistemi za analizo sentimenta na naši problemski domeni. Na kratko smo jih opisali v poglavju 2.7. Pri sistemu VADER in storitvah Indico smo uporabili notranje prečno preverjanje, pri vseh ostalih storitvah pa 10-kratno prečno preverjanje, kjer smo uporabili iste učne in testne množice, kot pri gradnji naših modelov v prejšnjem poglavju. To nam je omogočilo, da smo na koncu izvedli Wilcoxonove teste predznačenih rangov.

5.1 VADER

Sistem VADER, ki temelji na uporabi leksikona in modela pravil, ovrednoti vhodni komentar med -1 in 1 glede na sentiment. Vrednost -1 pomeni izjemno negativen komentar in 1 izjemno pozitiven. Ker takšno vrednotenje ni primerno za primerjavo ali komentar izraža navdušenje ali ne, moramo najprej poiskati primerno mejo, kjer lahko rezultate razdelimo v enega od pričakovanih razredov. Uporabili smo notranje prečno preverjanje. Na učni množici smo s sistemom VADER izračunali vrednosti sentimenta. Nato smo nad temi vrednostmi poiskali mejo, ki komentarje razdeli v enega izmed razredov in pri tem maksimizira vrednost mere F1. Vrednost mere F1 za to mejo smo nato izračunali na validacijski množici. Mejo smo dokončno dolo-

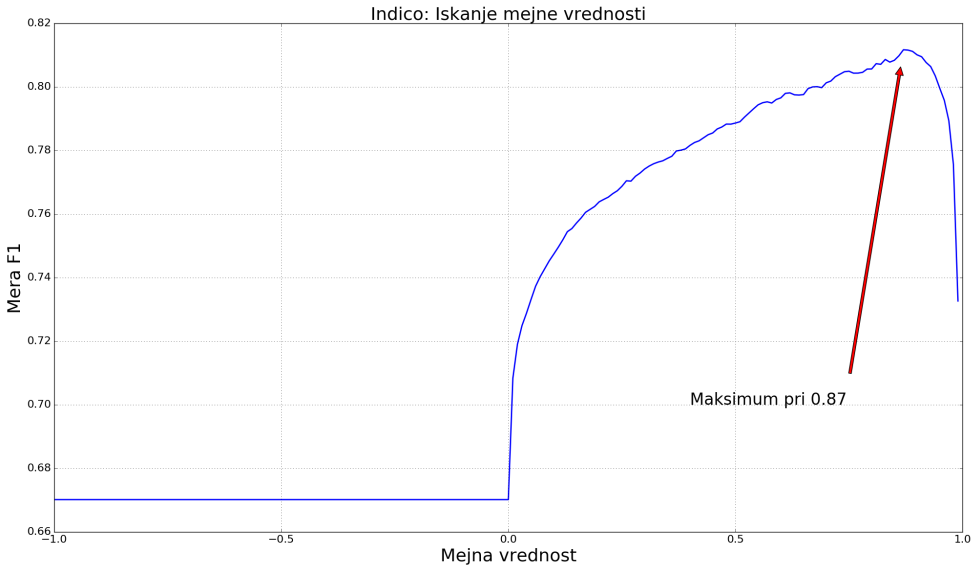
čili pri najvišji povprečni vrednosti mere F1 validacijskih množic. Slika 5.1 prikazuje vrednosti mere F1 na eni izmed učnih množic, pri različnih mejnih vrednostih. Najvišjo povprečno mero F1 smo dosegli pri mejni vrednosti 0.53.



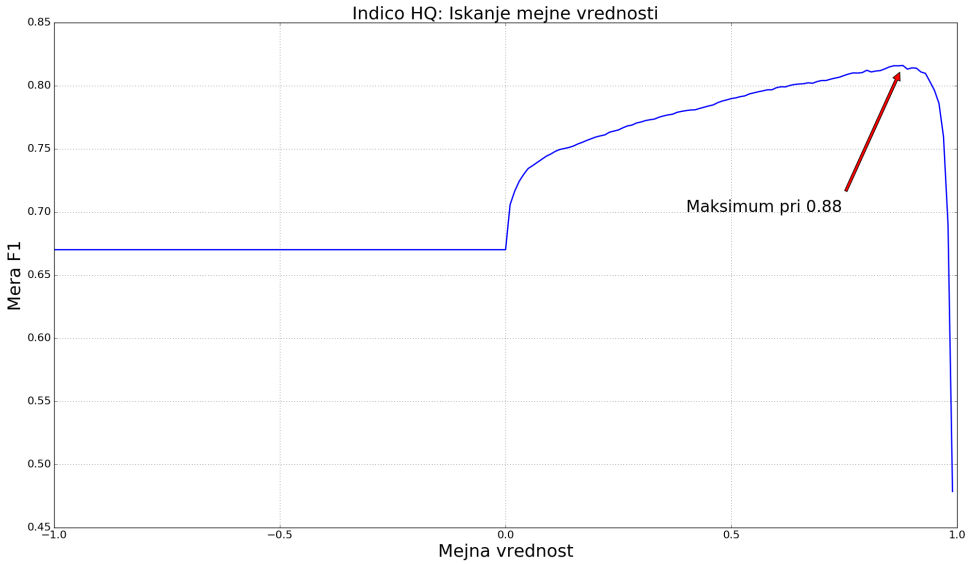
Slika 5.1: Iskanje mejne vrednosti na učni množici s pomočjo mere F1 pri sistemu VADER.

5.2 Indico

Indico ponuja dve različni metodi ocenjevanja, ki smo jih opisali v poglavju 2.7.2. Tudi ti dve metodi vrneta zvezne vrednosti na intervalu med 0 in 1 in tudi zanju smo ocenili mejo, ki kar najbolje označi komentarje. Postopek določanja meje je bil enak, kot pri sistemu VADER. Sliki 5.2 in 5.3 prikazujeta vrednosti mere F1 na eni izmed učnih množic, pri različnih mejnih vrednostih. Najboljša mejna vrednost za metodo Indico je bila v povprečju pri 0.87 in za metodo Indico HQ pri 0.88.



Slika 5.2: Iskanje mejne vrednosti na učni množici s pomočjo mere F1 pri storitvi Indico.



Slika 5.3: Iskanje mejne vrednosti na učni množici s pomočjo mere F1 pri storitvi Indico HQ.

5.3 Ostali sistemi

Google Prediction API, Amazon ML in BigML ponujajo izredno preprosto grajenje in evalvacijo modelov nadzorovanega učenja. Pri tem smo morali podati le označene komentarje, vsi ostali procesi, od obdelave besedil do izbora klasifikatorjev, pa so stekli avtomatsko in na njih nismo imeli neposrednega vpliva.

5.4 Rezultati

V tabeli 5.1 so prikazani rezultati izbranih sistemov in storitev. Statističnih razlik med rezultati Amazon ML, Google Prediction API in BigML ni bilo. Statistično pomembne razlike so bile med rezultati sistema VADER in Indico ($W = 0$, $p = 0.01$), ter Vader in Indico HQ ($W = 0$, $p = 0.01$). Prav tako so bile statistično pomembne razlike med rezultati Amazon ML, Google Prediction API, BigML in sistemi VADER, Indico, Indico HQ ($W = 0$, $p = 0.01$ pri vseh testih). Pri primerjavi z našimi sistemi so bile statistične razlike pri:

Google Prediction API: vsi nabori atributov, ki so uporabljali logistično regresijo in metodo podpornih vektorjev, so bili boljši ($W = 0$, $p = 0.01$). Nabor Lex pri uporabi naivnega Bayesa pa se je izkazal za slabšega ($W = 8$, $p = 0.05$).

BigML: tudi tu so bili boljši vsi nabori atributov, ki so uporabljali logistično regresijo in metodo podpornih vektorjev ($W = 0$, $p = 0.01$).

Amazon ML: nabori atributov Spell, Vader, All in Lex so se izkazali za slabše pri uporabi naivnega Bayesa (Spell: $W = 8$, $p = 0.05$; Vader: $W = 3$, $p = 0.01$; All: $W = 6$, $p = 0.05$; Lex: $W = 7$, $p = 0.05$). Vsi nabori atributov, ki so uporabljali logistično regresijo in metodo podpornih vektorjev, pa so bili boljši ($W = 0$, $p = 0.01$).

Indico, Indico HQ in VADER: vsi nabori atributov so bili boljši, ne glede na izbran klasifikator ($W = 0$, $p = 0.01$).

Sistem	Priklic	Preciznost	F1
VADER	0.84	0.71	0.77
Indico	0.91	0.70	0.78
Indico HQ	0.90	0.70	0.78
Google Prediction API	0.83	0.87	0.85
Amazon ML	0.82	0.88	0.85
BigML	0.85	0.83	0.84

Tabela 5.1: Rezultati primerjave z drugimi sistemi

Poglavje 6

Zaključek

V diplomski nalogi smo v prvem delu pregledali področje analize sentimenta in opisali pristope, ki se trenutno uporabljajo. Nato smo v drugem delu najprej analizirali komentarje iz spletne trgovine Google Play in nato razvili sistem, ki zna prepoznati tiste komentarje, ki izražajo navdušenje. Pri razvoju sistema smo se odločili za pristop z nadzorovanim učenjem. Razvili smo 8 različnih procesov za obdelavo vhodnega niza in s tem pridobili 8 različnih naborov atributov. Vsi sistemi so imeli skupen osnoven proces obdelave vhodnega niza. Ta je zajemal:

1. Razvoj sistema za razčlenjevanje vhodnega niza, ki je bil primeren za področje analize sentimenta;

2. Razvoj metode za razpoznavanje besed, ki obrnejo polarnost drugim stavčnim zvezam v besedilu;

3. Generacijo različnih n-gramov, ki so ohranili informacijo o sosledju besed in služili boljšemu pokrivanju pravilno in napačno zapisane besede;

4. Pretvorba niza žetonov v vrečo besed z uporabo binarne frekvence besed;

5. Dodajanje dveh umetnih žetonov v vrečo besed. To sta bila ocena komentarja in oblika zapisa besed v komentarju.

Šest sistemov je tudi zajemalo eno od naslednjih dodatnih metod:

1. Korenjenje nizov vhodnega komentarja;
2. Dodajanje oblikoslovnih oznak vhodnim komentarjem;
3. Popravljanje črkovanja v vhodnih komentarjih;
4. Dodajanje informacije o lokaciji besede v komentarju;
5. Dodajanje umetnega žetona o prisotnosti negativnih in pozitivnih besed s pomočjo mnenjskega leksikona;
6. Dodajanje dveh umetnih žetonov o negativnosti in pozitivnosti komentarja, ki smo jih pridobili s pomočjo sistema VADER.

Eden od sistemov je zajemal vse dodatne metode in eden nobene. Vse metode so bile smiselno vpete na neki stopnji pri osnovnem procesu obdelave besedila. Nato smo vhodne podatke vektorizirali in izvedli notranje prečno preverjanje. Izbor najboljših atributov bil izveden s pomočjo χ^2 metode. Pri metodi podpornih vektorjev in logistične regresije smo uporabili tudi regularizacijo. Na koncu smo pokazali, da obstaja statistično pomembna razlika med naivnim Bayesom in ostalima klasifikatorjema. Prav tako so bile statistično pomembne razlike med:

1. Naborom POS in nabori LOC in All;
2. Naborom Stem in nabori Spell in Vader;
3. Naborom Base in nabori Lex, Spell in Vader.

V zaključku smo na naši problemski domeni primerjali še sistem VADER in storitve Indico, Amazon ML, BigML in Google Prediction API. Sistem VADER in storitev Indico sta dosegla precej nižjo natančnost napovedovanja. Razlogi za to so verjetno v tem, da nobeden od sistemov ni bil posebej razvit za našo problemsko domeno in smo jih morali dodatno kalibrirati. Pri tem je storitev Indico uporabljala svoje predhodno naučene klasifikatorje in

tudi sistem VADER je bil prilagojen drugi domeni vhodnih podatkov. Ostale storitve so uporabljale nadzorovano učenje in so klasifikatorje naučile na naši problemski domeni. Statističnih razlik med rezultati storitev Amazon ML, Google Prediction API in BigML ni bilo. Statistično pomembne razlike so bile med rezultati Amazon ML, Google Prediction API, BigML in sistemi VADER, Indico, Indico HQ. Prav tako so bile statistično pomembne razlike med sistemi VADER in Indico, ter VADER in Indico HQ. Obstajale so statistične razlike med vsemi našimi modeli, ki so uporabljali SVM in logistično regresijo in storitvami Amazon ML, Google Prediction API in BigML. Rezultati vseh naših modelov so bili statistično boljši od sistemov VADER, Indico in Indico HQ. Prav tako so obstajale statistične razlike med:

1. Naborom Lex pri uporabi naivnega Bayesa in storitvami Google Prediction API in Amazon ML;

2. Nabori Spell, Vader, All in Lex pri uporabi naivnega Bayesa in storitvijo Amazon ML.

Sistem za analizo sentimenta na komentarjih v trgovini Google Play, ki smo ga razvili, je uporaben za označevanje navdušujočih komentarjev, ki so pod različnimi mobilnimi aplikacijami. Ti označeni komentarji lahko služijo nadaljnji analizi. Tako so primerni za ocenjevanje spremembe sentimenta skozi čas pri posamezni aplikaciji in tudi za ocenjevanje potencialne rasti uporabniške baze posamezne aplikacije.

Slike

1.1 Prikaz komentarjev v trgovini Google Play 3

2.1 Sistem VADER 18
2.2 Storitev BigML in prikaz odločitvenega drevesa. 19

3.1 Prikaz komentarja v strukturirani obliki 22
3.2 Porazdelitev komentarjev glede na oceno 26

4.1 Diskriminantna hiperravnina pri SVM 46
4.2 Vrednost mere F1 za klasifikatorje pri različnih naborih atributov. 59
4.3 Vrednost mere F1 pri različnih regularizacijskih parametrih za logistično regresijo in SVM 60
4.4 Vrednost mere F1 na učni in testni množici pri uporabi regularizacije 61

5.1 Iskanje mejne vrednosti na učni množici pri sistemu Vader . . 64
5.2 Iskanje mejne vrednosti na učni množici pri storitvi Indico . . 65
5.3 Iskanje mejne vrednosti pri storitvi Indico HQ 65

Tabele

2.1 Iskane oblikoslovne oznake 13
2.2 Primerjava nekaterih znanih leksikonov. 16

3.1 Primerjava števila povedi vseh označenih komentarjev. 23
3.2 Najbolj uporabljene besede v komentarjih. 24
3.3 Najbolj pogosto napačno zapisane besede 25

4.1 Primer razčlenjevanja besedila na praznih znakih. 28
4.2 Sklop regularnih izrazov 29
4.3 Negativne besedne vrste, ki obrnejo polarnost. 30
4.4 Primer negacije niza žetonov 31
4.5 Primeri n-gramov 32
4.6 Primer 2-gramov na nivju besede 32
4.7 Pravila korenjenja Porterjevega algoritma 34
4.8 Primeri negativnih in pozitivnih korenjenih besed 34
4.9 Rezultati črkovalnih sistemov 37
4.10 Primeri istih besed z različnimi skladenjskimi oznakami in polarnostmi 38
4.11 Seznam atributov pri oblikoslovnem označevanju 39
4.12 Wilcoxonova tabela kritičnih vrednosti 51
4.13 Nabor osnovnih atributov 53
4.14 Primeri atributov pri uporabi naprednih metod. 54
4.15 Nabori različnih atributov 55
4.16 Povprečni rezultati klasifikatorjev preko vseh naborov atributov 57

4.17 Povprečni rezultati naborov atributov preko vseh klasifikatorjev 57

5.1 Rezultati primerjave z drugimi sistemi 67

Literatura

[1] Accuracy and precision. Dostopno na https://en.wikipedia.org/wiki/Accuracy_and_precision.

[2] Machine learning and text feature extraction. Dostopno na http://blog.christianperone.com/2011/09/machine-learning-text-feature-extraction-tf-idf-part-i/.

[3] Machine Learning Tutorial: The Naive Bayes Text Classifier. Dostopno na http://blog.christianperone.com/2011/09/machine-learning-text-feature-extraction-tf-idf-part-i/.

[4] Sentiment Symposium Tutorial. Dostopno na http://sentiment.christopherpotts.net/index.html.

[5] TF-IDF. Dostopno na https://en.wikipedia.org/wiki/Tf\T1\textendashidf.

[6] Wilcoxon signed-rank test. Dostopno na https://en.wikipedia.org/wiki/Wilcoxon_signed-rank_test.

[7] Thorsten Brants, Ashok C Popat, Peng Xu, Franz J Och, in Jeffrey Dean. Large language models in machine translation. Objavljeno v *In Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*. Citeseer, 2007.

[8] Christopher Manning Dan Jurafsky. Natural Language Processing Course. Dostopno na `https://class.coursera.org/nlp/lecture`.

[9] Vasileios Hatzivassiloglou in Janyce M Wiebe. Effects of adjective orientation and gradability on sentence subjectivity. Objavljeno v *Proceedings of the 18th conference on Computational linguistics-Volume 1*, strani 299–305. Association for Computational Linguistics, 2000.

[10] Graham Hole. Wilcoxon test handout. Dostopno na `http://users.sussex.ac.uk/~grahamh/RM1web/teaching08-RS.html`.

[11] Matthew Honnibal. Part of speech tagger in Python. Dostopno na `https://spacy.io/blog/part-of-speech-pos-tagger-in-python`.

[12] Clayton J Hutto in Eric Gilbert. Vader: A parsimonious rule-based model for sentiment analysis of social media text. Objavljeno v *Eighth International AAAI Conference on Weblogs and Social Media*, 2014.

[13] N. Indurkhya in F.J. Damerau. *Handbook of Natural Language Processing, Second Edition*. Chapman & Hall/CRC Machine Learning & Pattern Recognition. CRC Press, 2010. Dostopno na `https://books.google.si/books?id=nK-QYHZ0-_gC`.

[14] D. Jurafsky in J.H. Martin. Speech and Language Processing. 2015. Dostopno na `https://web.stanford.edu/~jurafsky/slp3/`.

[15] Mark D Kernighan, Kenneth W Church, in William A Gale. A spelling correction program based on a noisy channel model. Objavljeno v *Proceedings of the 13th conference on Computational linguistics-Volume 2*, strani 205–210. Association for Computational Linguistics, 1990.

[16] John Lafferty, Andrew McCallum, in Fernando CN Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. 2001.

[17] B. Liu. *Sentiment Analysis and Opinion Mining*. Synthesis digital library of engineering and computer science. Morgan & Claypool, 2012. Dostopno na `https://books.google.si/books?id=Gt8g72e6MuEC`.

[18] Bing Liu. *Web data mining: exploring hyperlinks, contents, and usage data*. Springer Science & Business Media, 2007.

[19] Bing Liu. Sentiment analysis: A multi-faceted problem. *IEEE Intelligent Systems*, 25(3):76–80, 2010.

[20] Christopher D Manning, Prabhakar Raghavan, Hinrich Schütze, et al. *Introduction to information retrieval*, del 1. Cambridge university press Cambridge, 2008.

[21] Bo Pang in Lillian Lee. Opinion mining and sentiment analysis. *Foundations and trends in information retrieval*, 2(1-2):1–135, 2008.

[22] Bo Pang, Lillian Lee, in Shivakumar Vaithyanathan. Thumbs up?: sentiment classification using machine learning techniques. Objavljeno v *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*, strani 79–86. Association for Computational Linguistics, 2002.

[23] Ana-Maria Popescu in Orena Etzioni. Extracting product features and opinions from reviews. Objavljeno v *Natural language processing and text mining*, strani 9–28. Springer, 2007.

[24] Kumar Ravi in Vadlamani Ravi. A survey on opinion mining and sentiment analysis: tasks, approaches and applications. *Knowledge-Based Systems*, 89:14–46, 2015.

[25] Peter D Turney. Thumbs up or thumbs down?: semantic orientation applied to unsupervised classification of reviews. Objavljeno v *Proceedings of the 40th annual meeting on association for computational linguistics*, strani 417–424. Association for Computational Linguistics, 2002.